SynthWorks
VHDL Training Experts

RandomPkg Quick Ref

1. Package References
Using RandomPkg requires the following references:

```
library osvvm ;
use osvvm.RandomPkg.all ;
```

2. Randomization Object
Randomization requires a variable of protected type RandomPType.

```
process
  variable RV : RandomPType ;
```

3. Initializing the Seed
Initialize the internal seed with InitSeed:

```
RV.InitSeed(RV'instance_name) ;
```

4. Randomization with a Range
Randomly generate a value in a range, here 1 to 15.

```
DataInt := RV.RandInt(1, 15) ;
```

Overloading for std_logic_vector, unsigned and signed require a size parameter (last). Overloading for real produces an open range.

```
DataReal := RV.RandReal(5.5, 99.3) ;
DataSlv8 := RV.RandSlv(1, 15, 8) ;
DataUv8  := RV.RandUnsigned(1, 15, 8) ;
DataSv8  := RV.RandSigned(-8, 7, 8) ;
```

Overloading for time requires specification of time units.

```
dTime := RV.RandTime(5 ns, 10 ns, 1 ns) ;
```

5. Randomization with a Range + Exclude
Randomly generate a value within a range (here 1 to 15) and exclude specified values (integer_vector), here 3, 5, and 7.

```
dInt := RV.RandInt(1, 15, (3, 5, 7));
```

Exclude values are supported for std_logic_vector, unsigned, and signed, and are not supported for type time or real.

```
Slv8 := RV.RandSlv(1, 15, (3, 5, 7), 8);
```

6. Randomization of a Set
Randomly generate a set (integer_vector), here 1, 3, 5, 7, and 9.

```
dInt := RV.RandInt( (1,3,5,7,9) ) ;
```

Overloading for std_logic_vector, unsigned and signed require a size parameter (last). Also supported for type time and real.

```
DataSlv8 := RV.RandSlv( (1,3,5,7,9), 8) ;
dTime := RV.RandTime((1 ns, 3 ns, 5 ns));
dReal := RV.RandReal( (1.1, 1.4, 2.1) );
```

7. Randomization of a Set + Exclude
Randomly generate a set (integer_vector), here 1, 3, 5, 7, and 9 and exclude specified values (integer_vector), here 3 and 7.

```
dInt := RV.RandInt((1,3,5,7,9), (3,7)) ;
```

Overloading for std_logic_vector, unsigned and signed require a size parameter (last). Also supported for type time and real.

8. Larger than Integer Ranges
For std_logic_vector, unsigned, and signed, the following forms of functions generate values larger than 32 bits. In the first example, 40 is the size of the vector. It the next examples, MIN_VAL_40 and MAX_VAL_40 are a 40 bit std_logic_vectors that specify the minimum and maximum values.

```
dSlv40 := RV.RandSlv(40) ;
dSlv40 := RV.RandSlv(MAX_VAL_V40) ;
dSlv40 := RV.RandSlv(MIN_V40,MAX_V40) ;
```

9. Weighted Randomization
Specify an array of values and weights and randomly select one of the values with a probability of the weight divide by the sum of weights. In the following the first parameter is the value and the second is the weight. Hence values are 1, 3, and 5 and the weights 7, 3, and 1. As a result, the probability of 1 is (7/10) or 70%, 3 is (2/10) or 20%, and 5 is 10 %. Generally it is easiest if the weights sum up to either 10 or 100.

```
dInt :=
   RV.DistValInt( ((1,7),(3,2),(5,1)) );
```

There is also a DistValSlv, DistValUnsigned, and DistValSigned. All of the functions also support an exclude vector.

10. Simple Weighted Randomization
Specify N weights as an integer_vector (weights 7, 2, and 1) and randomly select a value in the range of the weight parameter (for a literal, 0 to N-1 – here 0 to 2). The probability of each value being generated is the weight divide by the sum of weights. Here the probability of 0 is (7/10) or 70%, 1 is (2/10) or 20%, and 2 is 10 %. Generally it is easiest if the weights sum up to either 10 or 100.

```
dInt := RV.DistInt( (7,2,1) ) ;
```

There is also a DistSlv, DistUnsigned, and DistSigned. All of the functions also support an exclude vector.

11. Normal, aka Gaussian Distribution
A function normal receives a Mean (here 1.0) and Standard Deviation (here 0.025) and returns a random value with a normal or Gaussian distribution.

```
dReal := RV.normal(1.0, 0.025);
```

When the value is outside of the range specified by the optional minimum (here 0.9) or maximum (here 1.1), the value is re-randomized.

```
dReal := RV.normal(1.0, 0.025, 0.9, 1.1);
```

Normal is also overloaded to have integer return and minimum and maximum parameters as shown below.

```
dInt := RV.normal(100.0, 0.025, 90, 110);
```

12. Other Distributions
RandomPkg also supports Poisson, FavorSmall, FavorBig, and Uniform distributions. See RandomPkg Users Guide for details.

13. Randomize vectors of Integer, Real, Time
RandomPkg also supports RandIntV, RandRealV, and RandTimeV to randomly generate vectors of Integer, Real, and Time. See RandomPkg Users Guide for details.

14. Packages
The following files are required for randomization, RandomPkg.vhd, RandomBasePkg.vhd, and SortListPkg_int.vhd. Generally we create an OSVVM library and compile these files into it. Note that some of the simulator vendors have done this for you.

SynthWorks Design Inc.
VHDL Intro, RTL, and Verification Training
11898 SW 128th Ave. Tigard OR 97223 (800)-505-8435
http://www.SynthWorks.com jim@synthworks.com