



FREE eBook

LEARNING random

Free unaffiliated eBook created from
Stack Overflow contributors.

#random

Table of Contents

About.....	1
Chapter 1: Getting started with random.....	2
Remarks.....	2
Examples.....	2
Installation or Setup.....	2
Fisher-Yates shuffle.....	2
Credits.....	4

About

You can share this PDF with anyone you feel could benefit from it, downloaded the latest version from: [random](#)

It is an unofficial and free random ebook created for educational purposes. All the content is extracted from [Stack Overflow Documentation](#), which is written by many hardworking individuals at Stack Overflow. It is neither affiliated with Stack Overflow nor official random.

The content is released under Creative Commons BY-SA, and the list of contributors to each chapter are provided in the credits section at the end of this book. Images may be copyright of their respective owners unless otherwise specified. All trademarks and registered trademarks are the property of their respective company owners.

Use the content presented in this book at your own risk; it is not guaranteed to be correct nor accurate, please send your feedback and corrections to info@zzzprojects.com

Chapter 1: Getting started with random

Remarks

This section provides an overview of what random is, and why a developer might want to use it.

It should also mention any large subjects within random, and link out to the related topics. Since the Documentation for random is new, you may need to create initial versions of those related topics.

Examples

Installation or Setup

Detailed instructions on getting random set up or installed.

Fisher-Yates shuffle

Also known as the Knuth shuffle and the Durstenfeld-Fisher-Yates shuffle. This shuffle takes an array of n elements and shuffles it. The algorithm is truly random in that, after shuffling, each permutation of the array is equally likely.

In java:

```
public static void shuffle(E[] deck) {  
  
    //From the end, swap each card with a random card from the unswapped portion.  
    for(int i = deck.length - 1; i > 0; i--)  
    {  
        //Pick an element from [0,i], inclusive.  
        int chosenCard = (int) (Math.random() * (i + 1));  
  
        E temp = deck[i];  
        deck[i] = deck[chosenCard];  
        deck[chosenCard] = temp;  
    }  
}
```

Please note: it is necessary that the replacement element come from $[0,i]$ inclusive and not $[0,i)$ exclusive: Otherwise, permutations of the array where elements remain in place are impossible, which is not truly random.

Assuming assuming random numbers take $O(1)$ to generate, the algorithm operates in place and takes $O(n)$ time and space. An array shuffled this way can be used to retrieve non-repeating elements in $O(1)$ amortized time per element.

```
E[] deck;  
int drawIndex;
```

```
//Elements are taken from an index that advances.
public E drawUniqueCard()
{
    //Once all cards have been drawn, reshuffle the deck and draw from the top.
    if(drawIndex == deck.length)
    {
        shuffle(deck);
        drawIndex = 0;
    }
    //Pull the next card off the deck.
    return deck[drawIndex++];
}
```

Read [Getting started with random](https://riptutorial.com/random/topic/9484/getting-started-with-random) online: <https://riptutorial.com/random/topic/9484/getting-started-with-random>

Credits

S. No	Chapters	Contributors
1	Getting started with random	Community , Mauve Ranger