



ORACLE

Academy



Java Foundations

4-4

Clase Random

ORACLE
Academy



Objetivos

- En esta lección se abordan los siguientes objetivos:
 - Describir el objetivo y los usos de los números aleatorios en la programación de Java
 - Identificar los métodos de la clase Random que obtienen números aleatorios
 - Obtener números aleatorios en un rango de números
 - Comprender el objetivo valor inicial del número aleatorio



Objetivo de la generación de números aleatorios en Java

- A menudo, una aplicación de software necesita realizar una tarea basada en un valor obtenido al azar
- Hay aplicaciones que necesitan la generación de números aleatorios
- Veamos algunas aplicaciones que utilizan la generación de números aleatorios



Aplicaciones basadas en generación de números aleatorios

- Una aplicación de cartas necesita barajar las cartas aleatoriamente y, a continuación, distribuirlas a los jugadores



- Una aplicación de lotería necesita un número generado aleatoriamente que se base en un algoritmo El jugador gana si su número coincide con el número generado de forma aleatoria



Generación de números aleatorios en Java

- En las lecciones anteriores ha visto que Java incluye una variedad de clases que soportan casi todas las funciones básicas de desarrollo de aplicaciones
- Por ejemplo:
 - La clase String proporciona la capacidad de manipular cadenas
 - La clase Scanner proporciona la capacidad de obtener una entrada desde la consola
- Otra clase importante en Java es Random, que se utiliza para obtener números aleatorios

¿Qué es la clase Random en Java?

- En Java, la clase Random se utiliza para obtener números aleatorios
- La clase está en el paquete java.util
- Contiene varios métodos que devuelven valores de tipo integer, double, boolean, float y long obtenidos aleatoriamente

¿Cómo se utiliza la clase Random en un programa Java?

- Importe la clase Random del paquete java.util
- Cree una instancia de la clase Random, del siguiente modo:

```
import java.util.Random;
```

Sentencia import para importar la clase Random del paquete java.util

```
public class RandomIntNums {  
    public static void main(String[] args) {  
        Random rndNumber = new Random();  
    } //end method main  
} //end class RandomIntNums
```

Crea una instancia de la clase Random, rndNumber

Métodos que proporciona la clase Random

- Puede obtener valores aleatorios mediante la llamada a los siguientes métodos proporcionados en la clase Random:

Método	Produce
<code>boolean</code> nextBoolean();	Un valor true o false
<code>int</code> nextInt()	Un valor entero entre Integer.MIN_VALUE e Integer.MAX_VALUE
<code>long</code> nextLong()	Un valor entero largo entre Long.MIN_VALUE y Long.MAX_VALUE
<code>float</code> nextFloat()	Un número decimal entre 0,0 (incluido) y 1,0 (excluido)
<code>double</code> nextDouble()	Un número decimal entre 0,0 (incluido) y 1,0 (excluido)

¿Cómo se obtiene un número aleatorio?

- Puede obtener un número aleatorio de tipo entero utilizando el método `nextInt`
- Por ejemplo:

```
import java.util.Random;
public class RandomNum {
    public static void main(String[] args) {
        Random rndNum = new Random();
        int randomNum = rndNum.nextInt();
        System.out.println("Random Number: " + randomNum);
    } //end method main
} //end class RandomNum
```

- Resultado:

Random Number: 1660093261

¿Cómo se obtiene una serie de números aleatorios?

- Puede obtener una serie de números aleatorios mediante la llamada al método `nextInt`
- Por ejemplo:

```
public class RandomNumSeries {  
    public static void main(String[] args) {  
        Random num = new Random();  
        System.out.println("Random Number 1: " + num.nextInt());  
        System.out.println("Random Number 2: " + num.nextInt());  
        System.out.println("Random Number 3: " + num.nextInt());  
        System.out.println("Random Number 4: " + num.nextInt());  
        System.out.println("Random Number 5: " + num.nextInt());  
    } //end method main  
} //end class RandomNumSeries
```

`nextInt()` se llama 5 veces y se generan 5 números aleatorios





Generación de números aleatorios de tipo Double

- Puede obtener números aleatorios de tipo double mediante el método `nextDouble` del siguiente modo:

```
public class RandomDouble {  
    public static void main(String[] args) {  
        Random num = new Random();  
        double randomDouble = num.nextDouble();  
        System.out.println("Random Number: " + randomDouble);  
    } //end method main  
} //end class RandomDouble
```

- En este ejemplo, el método `nextDouble` devuelve números del tipo double en el rango de 0.0 a 1.0

Ejercicio 1

- Cree un nuevo proyecto y agréguele el archivo `FlipCoin.java`
- Examine `FlipCoin.java`:
 - Ejecute el siguiente programa y observe el número aleatorio que se ha generado
 - Si es menor de 0,5, registre el resultado como “heads”; de lo contrario, registre el resultado como “tails”
 - Repita la operación varias veces



Generación de números aleatorios en un rango de números

- Hasta el momento, ha generado un número aleatorio en el rango de un tipo de dato entero
- A veces, puede que desee restringir el rango de números que se pueden generar
- Para implantarlo, puede utilizar otra versión del método `nextInt`:
 - `nextInt(int maxValue);`
 - El argumento determina el número entero más alto que se puede obtener con el método `nextInt()`
 - Puede obtener números positivos aleatorios del 0 (incluido) a un máximo que elija (excluido)

Generación de números aleatorios en un rango de números: Ejemplo

- A continuación se muestra un ejemplo que obtiene números aleatorios en el rango de 0 a 19:

```
public class RandomNumRange {  
    public static void main(String[] args) {  
        Random num = new Random();  
        int randomnum = num.nextInt(20);  
        System.out.println("Random Number: " + randomnum);  
    } //end method main  
} //end class RandomNumRange
```

Generación de un rango a partir del 1

- Para especificar un rango que empiece por 1, sume 1 al resultado del método `nextInt()`
- Por ejemplo, para elegir un número entre 1 y 40 inclusive, sume 1 al resultado:

```
Random rand = new Random();  
int randomnum = rand.nextInt(40) + 1;
```

Generación de un rango a partir de un número mayor que 1

- Si el rango empieza desde un número mayor que 1:
 - Reste el número de inicio al número de límite superior y, a continuación, sume 1
 - Sume el número de inicio al resultado del método `nextInt()`.
- Por ejemplo, para elegir un número entre 5 y 35 inclusive:
 - El número de límite superior será $35 - 5 + 1 = 31$ y se debe sumar 5 al resultado:

```
Random rand = new Random();  
int randomnum = rand.nextInt(31) + 5;
```

Programa para aplicación de lotería



```
public class Lottery {  
  
    public static void main(String[] args) {  
  
        Scanner numberScanner = new Scanner(System.in);  
        System.out.print("Enter a number between 1 and 10: ");  
        int userNum = numberScanner.nextInt();  
        Random rnd = new Random();  
        int winningNum = rnd.nextInt(10) + 1;  
        System.out.println("Your Number: " + userNum);  
        System.out.println("The winning number is: " + winningNum);  
    } //end method main  
  
} //end class RandomNumRange
```

Ejercicio 2

- Cree un nuevo proyecto y agréguele el archivo `RockPaperScissor.java`
- Examine `RockPaperScissor.java`
 - Realice lo siguiente:
 - Simule el juego piedra-papel-tijera generando un número entero aleatorio en el rango de 0 a 3
 - Compare el número generado con los números siguientes:
 - Si número=0: "piedra"
 - Si número=1: "papel"
 - Si número=2: "tijera"
 - Registre el resultado y repita el proceso varias veces

¿Se genera el mismo número aleatorio cada vez?

- Al ejecutar los ejemplos anteriores varias veces, observe que la secuencia de números aleatorios es diferente cada vez
- En ocasiones, puede que deba generar la misma secuencia de números aleatorios cada vez



¿Qué es el valor inicial de un número aleatorio?

- Para ello, utilice un valor constante denominado inicial
- Al crear una instancia de la clase Random, transfiere un entero constante para especificar el valor inicial

```
Random rndNumbers = new Random(20L);
```



Elemento inicial

- Puede cambiar el valor inicial mediante la llamada al método `setSeed()`
- Cada vez que transfiere el mismo valor inicial, se devuelve la misma secuencia aleatoria

Obtención de una secuencia aleatoria mediante un elemento inicial: Ejemplo

```
public static void main(String[] args) {  
    Random rand = new Random(20L);  
    System.out.println("Random Number 1: " + rand.nextInt(100));  
    System.out.println("Random Number 2: " + rand.nextInt(100));  
    System.out.println("Random Number 3: " + rand.nextInt(100));  
  
    System.out.println("Changing seed to change to sequence");  
    rand.setSeed(5L);  
    System.out.println("Random Number 4: " + rand.nextInt(100));  
    System.out.println("Random Number 5: " + rand.nextInt(100));  
    System.out.println("Random Number 6: " + rand.nextInt(100));  
  
    System.out.println("Setting seed 20 produce previous sequence");  
    rand.setSeed(20L);  
    System.out.println("Random Number 7: " + rand.nextInt(100));  
    System.out.println("Random Number 8: " + rand.nextInt(100));  
    System.out.println("Random Number 9: " + rand.nextInt(100));  
} //end method main
```

Alternativa a la clase Random: Math.random()

- El método **java.lang.Math.random()** devuelve un número pseudoaleatorio (tipo double) mayor o igual que 0,0 y menor que 1,0
- Cuando se llama a este método, crea un nuevo generador de números pseudoaleatorios, de la misma forma que si se utilizara la clase Random
- Introduzca el código de la nota de la diapositiva que aparece a continuación
- Ejecute el código varias veces para ver cómo difieren los resultados

Resumen

- En esta lección, debe haber aprendido lo siguiente:
 - Describir el objetivo y los usos de los números aleatorios en la programación de Java
 - Identificar los métodos de la clase Random que obtienen números aleatorios
 - Obtener números aleatorios en un rango de números
 - Comprender el objetivo valor inicial del número aleatorio





ORACLE

Academy

