En este tutorial veremos cómo crear un programa en Java que genere una contraseña aleatoria. Para este ejemplo se utilizó el IDE IntelliJ IDEA, pero puedes utilizar el de tu preferencia.

Generador de contraseñas en Java

El algoritmo en el que se basa este programa es el siguiente:

1.- Crear un programa que genere una contraseña tomando en cuenta los siguientes requerimientos:

- Debe tener 8 caracteres
- Debe tener 2 letras mayúsculas
- Debe tener 2 caracteres especiales
- Debe tener 4 caracteres alfanuméricos aleatorios
- Imprimir el resultado en pantalla

2.- Opcional a esto, mostrar un menú para poder elegir si se requiere una contraseña nueva.

El código utilizado para el punto 1 es el siguiente:

```
import java.util.Random;

public class PassGen {
    public static void main(String[] args) {
        String randomCode = generateRandomCode();
        System.out.println("Password generado: " +
randomCode);
    }
```

```
    public static String generateRandomCode() {
        Random random = new Random();
        StringBuilder codeBuilder = new StringBuilder();

        // Generar 2 letras mayúsculas
        for (int i = 0; i < 2; i++) {
            char randomUppercaseLetter = (char)
(random.nextInt(26) + 'A');
            codeBuilder.append(randomUppercaseLetter);
        }

        // Generar 2 caracteres especiales
        String specialCharacters = "!@#$%^&*()";
        for (int i = 0; i < 2; i++) {
            int randomIndex =
random.nextInt(specialCharacters.length());
            char randomSpecialChar =
specialCharacters.charAt(randomIndex);
            codeBuilder.append(randomSpecialChar);
        }

        // Generar 4 caracteres alfanuméricos aleatorios
        String alphanumeric =
"abcdefghijklmnopqrstuvwxyz0123456789";
        for (int i = 0; i < 4; i++) {
            int randomIndex =
random.nextInt(alphanumeric.length());
            char randomChar =
alphanumeric.charAt(randomIndex);
            codeBuilder.append(randomChar);
```

```
        }

        // Mezclar los caracteres aleatoriamente
        for (int i = codeBuilder.length() - 1; i > 0; i--) {
            int j = random.nextInt(i + 1);
            char temp = codeBuilder.charAt(i);
            codeBuilder.setCharAt(i, codeBuilder.charAt(j));
            codeBuilder.setCharAt(j, temp);
        }

        return codeBuilder.toString();
    }
}
```

Se importa la clase *Random* del paquete *java.util,* mediante *java.util.Random* para trabajar con caracteres aleatorios.

Después se crea una clase llamada *RandonCodeGenerator*, que contienen un método denominado *main*, donde se indica la ejecución del programa. En el método se llama al otro método *generateRandomCode()* con el fin de generar un código aleatorio e imprimirlo en la consola.

Posteriormente, se crea un método llamado *generateRandomCode()*, que devuleve una cadena de caracteres de tipo string y se crea una instancia de la case *Random* para generar números aleatorios y un objeto *StringBuilder* para construir el código aleatorio.

Luego tenemos un bucle *for*, en el que se generan dos letras mayúsculas de forma aleatoria. Se utiliza el método *nextInt(26)* para generar un número aleatorio entre 0 y 25 y sumarlo al valor ASCII de *A* para obtener un número aleatorio en el rango de las letras mayúculas.

Después se convierte el número en un caracter y se añade al objeto *codeBuilder*.

Ahora se genera una cadena de caracteres *specialCharacters*, que contiene los caracteres especiales permitidos. En un bucle for se generan dos índices aleatorios dentro del rango de la longitud de *specialCharacters*. Después se seleccionan los caracteres especiales correspondientes a los índices generados y se añaden al objeto *codeBuilder*.

Después, se define una cadena de caracteres *alphanumeric*, que contiene todas las letras mayúsculas y dígitos del 0 al 9. En un bucle for se generan cuatro índices aleatorios del rango de la longitud de *alphanumeric*. Después se seleccionan los caracteres alfanuméricos correspondientes a los índices generados y se añaden al objeto *codeBuilder*.

Finalmente, en un bucle for se mezclan los caracteres en el objeto *codeBuilder* de forma aleatoria. Se comienza desde el último caracter y en cada iteración, se genera un índice aleatorio *j* dentro del rango de los índices no procesados. Se intercambia el caracter en la posición *i* con el caracter en la posición *j*.

Al final, se retorna el código aleatorio como una cadena de caracteres llamando al método *tostring()* del objeto *codeBuilder*.

Añadir un menú al programa para poder seguir generando contraseñas

Para cumplir con el punto número 2, agregamos una sentencia while para crear un menú que al mostrar una contraseña, pregunte si se requiere generar otra, al presionar Si, se genera un nuevo password, de lo contrario, con cualquier otra tecla, se termina el programa.

El código creado es el siguiente:

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

```
public class PassGen {
public static void main(String[] args) {
Scanner scanner = new Scanner(System.in);
Random random = new Random();

boolean generateMoreCodes = true;
while (generateMoreCodes) {
String randomCode = generateRandomCode(random);
System.out.println(«Password generado: » + randomCode);

System.out.print(«¿Desea generar otro password? (Sí/No): «);
String input = scanner.nextLine();

if (!input.equalsIgnoreCase(«s») && !input.equalsIgnoreCase(«Si»)) {
generateMoreCodes = false;
}
}

scanner.close();
}

public static String generateRandomCode(Random random) {
StringBuilder codeBuilder = new StringBuilder();

// Generar 2 letras mayúsculas
for (int i = 0; i < 2; i++) {
char randomUppercaseLetter = (char) (random.nextInt(26) + ‘A’);
codeBuilder.append(randomUppercaseLetter);
}

// Generar 2 caracteres especiales
String specialCharacters = «!@#$%^&*()»;
```

```
for (int i = 0; i < 2; i++) {
int randomIndex = random.nextInt(specialCharacters.length());
char randomSpecialChar = specialCharacters.charAt(randomIndex);
codeBuilder.append(randomSpecialChar);
}

// Generar 4 caracteres alfanuméricos aleatorios
String alphanumeric = «abcdefghijklmnopqrstuvwxyz0123456789»;
for (int i = 0; i < 4; i++) {
int randomIndex = random.nextInt(alphanumeric.length());
char randomChar = alphanumeric.charAt(randomIndex);
codeBuilder.append(randomChar);
}

// Mezclar los caracteres aleatoriamente
for (int i = codeBuilder.length() – 1; i > 0; i–) {
int j = random.nextInt(i + 1);
char temp = codeBuilder.charAt(i);
codeBuilder.setCharAt(i, codeBuilder.charAt(j));
codeBuilder.setCharAt(j, temp);
}

return codeBuilder.toString();
}
}
```

Aquí se utiliza el bucle while para que se mantenga en ejecución el programa mientras la respuesta del usuario sea si o s (sin importar mayúsculas o minúsculas). Si la respuesta es diferente a esto, se rompe el bucle y finaliza el programa. Se debe notar que se importó la *Scanner de java.util* para poder capturar lo que el usuario ingresa en el teclado.

Si requieres algún programa en específico, no dudes en contactarnos aquí para una cotización.

Si tienes alguna duda o comentario, déjalo aquí abajo!