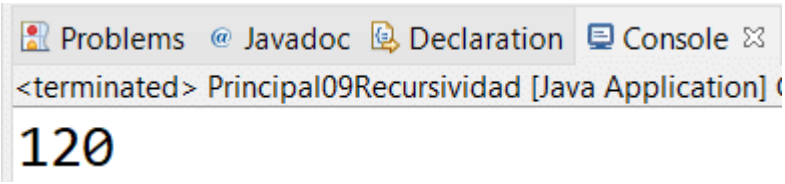
Java Recursividad y ficheros . Este artículo aborda el concepto de recursividad en Java y nos muestra como se puede usar de una forma sencilla para gestionar el manejo de ficheros y carpetas usando el api más clásico que soporta el lenguaje.

Java Recursividad

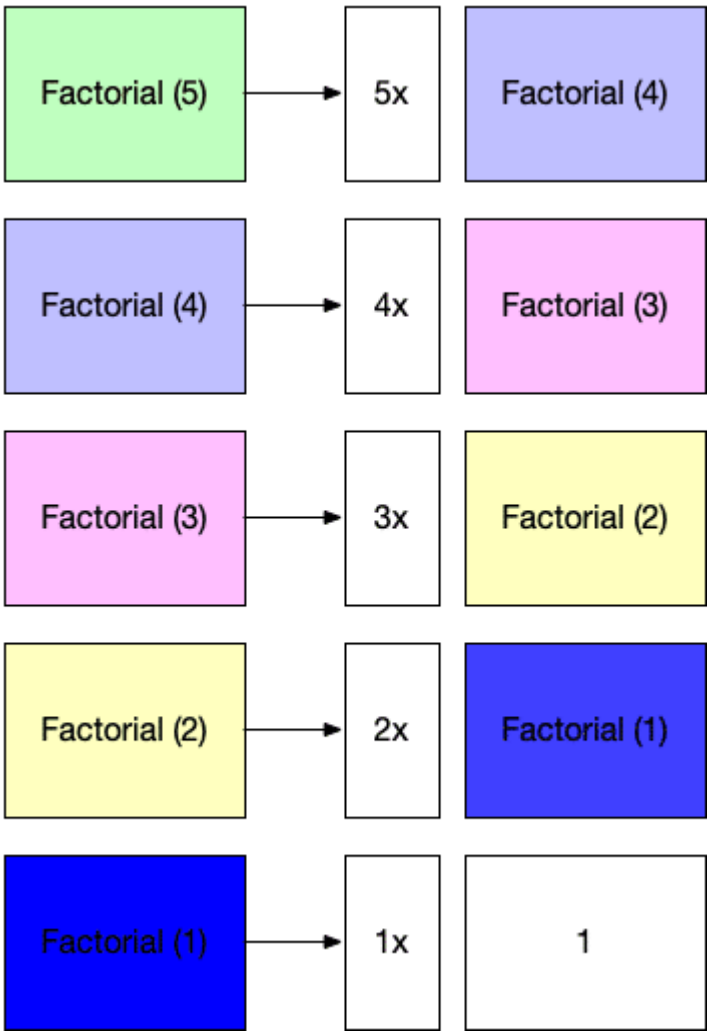
La recursividad esta muy ligada a la repetición de una misma funcionalidad n veces . Eso sí en vez de construir a través de un bucle se construye invocandose una función a si misma varias veces hasta que se cumple una condición determinada. Por ejemplo imaginemonos que queremos calcular el factorial de un número. La forma más sencilla de realizar esta operación es a través de un bucle for. Recordemos que el factorial de 5 es 5x4x3x2x1

```
public class Principal09Recursividad {

	public static void main(String[] args) {
		System.out.println(factorial(5));
	}
	public static int factorial(int numero) {
		//5x4x3x2x1
		int factorial=1;
		for (int i=numero;i>0;i--) {
			factorial=factorial*i;
		}
		return factorial;
	}
}
```

De esta manera nosotros invocamos la función factorial y calculamos el factorial de 5 que es 120 podemos ver el resultado impreso en la consola.



De igual manera podemos realizar esa operación a través de una función recursiva es decir una función que se invoca a si misma para realizar el calculo del factorial , pero con otro valor de parámetro.



Vamos a verlo:

```
public static int factorialRecursivo(int numero) {
        //5x4x3x2x1
```

```
            // 5x factorial(4)
            //5x4xfactorial(3)
            if (numero==1) {
                    return numero;
            }
            return numero*factorialRecursivo(numero-1);
    }
```

En este caso estamos ante una situación cuyo enfoque es recursivo pero si la ejecutamos el resultado es idéntico.

Java Recursividad vs Bucles

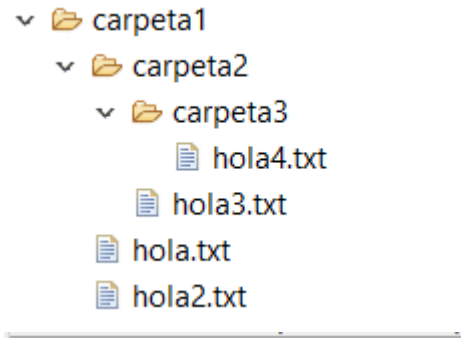
En muchas ocasiones los alumnos me preguntan porque usar la recursividad si podemos usar un bucle. La respuesta es bastante sencilla , en los casos básicos como el del factorial no tenemos problema ya que conocemos el tope o límite para el cual tenemos que hacer el calculo .En nuestro caso es la situación en la cual i>0 para que la multiplicación se produzca de forma correcta y no terminemos multiplicando por cero.

Ahora bien existen situaciones en las que no conocemos de entrada el tope. Por ejemplo imaginémonos que deseamos ver los ficheros que tiene una carpeta con todas sus subcarpetas y ficheros . Esto no tiene un limite claro puedo tener una sola carpeta y ficheros dentro o puede tener un arbol de carpetas y ficheros muy amplio. Es en este caso en donde la recursivdad nos puede ser muy útil ya que nos permite invocar la función de forma reiterada mientras se cumpla una condición .Vamos a verlo:

```
package com.arquitecturajava;

import java.io.File;
import java.io.IOException;

public class Principal07 {
```

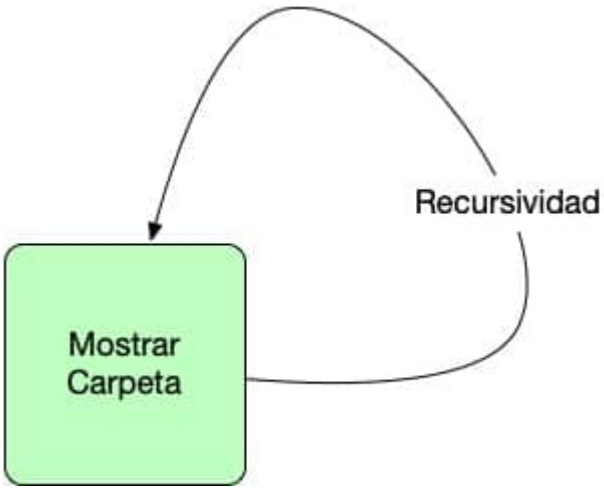
```
	public static void main(String[] args) throws IOException {

		File fichero = new File("carpeta1");

		mostrarCarpeta(fichero);
	}

	// una funcion recursiva que se llame a si misma
	public static void mostrarCarpeta(File fichero) {

		if (fichero.isDirectory()) {

			File[] lista = fichero.listFiles();

			for (int i = 0; i < lista.length; i++) {

System.out.println(lista[i].getName());

				if (lista[i].isDirectory()) {
					mostrarCarpeta(lista[i]);
				}
			}
		}

	}

}
```

En este caso vamos a usar esta funcionalidad para recorrer la siguiente estructura de carpetas:




```
carpeta2
carpeta3
hola4.txt
hola3.txt
hola.txt
hola2.txt
```

Como podemos ver se muestra de forma recursiva el contenido de cada una de las subcarpetas de la carpeta 2. Hemos invocado de forma recursiva la función mostrarCarpeta(fichero).



Otros artículos relacionados

- File to String Java 8 y manejo de ficheros
- Java 8 Files Walk y Recursividad
- Java Stream File y manejo de ficheros
- https://cursos.arquitecturajava.com/p/java-apis