

Java Predicate Not y JDK 11 .Negar un predicado es una de las operaciones más sencillas que normalmente queremos hacer con programación funcional . Es decir nosotros podemos tener una lista de objetos convertirlos en un Stream y aplicar un **Predicate** , eso sí queremos que el Predicate se aplique a la inversa. Vamos a verlo utilizando la clase Persona.

```
package com.arquitecturajava;
```

```
public class Persona {
```

```
    private String nombre;
```

```
    private String apellidos;
```

```
    private int edad;
```

```
    public String getNombre() {
```

```
        return nombre;
```

```
    }
```

```
    public void setNombre(String nombre) {
```

```
        this.nombre = nombre;
```

```
    }
```

```
    public String getApellidos() {
```

```
        return apellidos;
```

```
    }
```

```
    public void setApellidos(String apellidos) {
```

```
        this.apellidos = apellidos;
```

```
    }
```

```
    public int getEdad() {
```

```
        return edad;
```

```
    }
```

```
    public void setEdad(int edad) {
```

```
        this.edad = edad;
```

```
    }
```

```
    public Persona(String nombre, String apellidos, int edad) {
```

```

        super();
        this.nombre = nombre;
        this.apellidos = apellidos;
        this.edad = edad;
    }
    public boolean estaJubilado() {
        return this.edad>65;
    }
    @Override
    public String toString() {
        return "Persona [nombre=" + nombre + ", apellidos=" +
apellidos + ", edad=" + edad + "];"
    }
}

```

Es momento de construir una lista y filtrar las personas que se han jubilado utilizando un método de referencia y el método filter.

```
package com.arquitecturajava;
```

```
import java.util.Arrays;
```

```
import java.util.List;
```

```
public class Principal2 {
```

```

    public static void main(String[] args) {
        Persona persona1= new Persona ("ana","sanchez",30);
        Persona persona2= new Persona ("pedro","gomez",70);
        Persona persona3= new Persona
("maria","fernandez",25);
        Persona persona4= new Persona ("david","perez",75);
        List<Persona> lista=

```

```

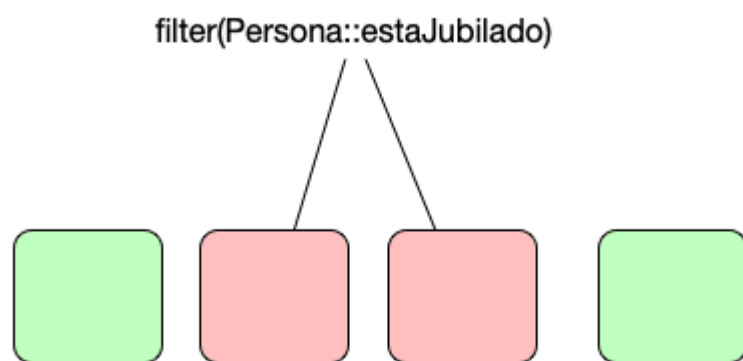
Arrays.asList(persona1,persona2,persona3,persona4);
lista.stream().filter(Persona::estaJubilado).forEach(System.out::println);

    }

}

```

Hasta aquí todo muy sencillo:



Si ejecutamos el código nos mostrará por la consola las Personas que estan jubiladas.

```

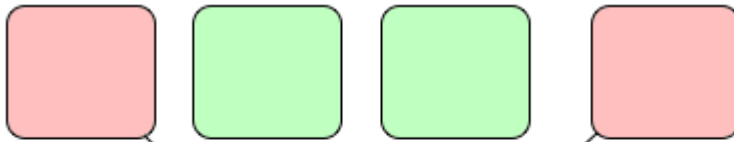
Persona [nombre=pedro, apellidos=gomez, edad=70]
Persona [nombre=david, apellidos=perez, edad=75]

```

Predicate Not y Java 11

A veces no necesitamos utilizar directamente el predicado sino que necesitamos utilizar una negación de este .

Para ello podemos usar el nuevo método not del interface Predicate en Java 11 que nos permitirá realizar dicha operación e imprimir en la consola aquellos que no estan jubilados .



`filter(Predicate.not(Persona::estaJubilado))`

Vamos a verlo:

```
package com.arquitecturajava;

import java.util.Arrays;
import java.util.List;
import java.util.function.Predicate;

public class Principal {

    public static void main(String[] args) {
        Persona persona1= new Persona ("ana","sanchez",30);
        Persona persona2= new Persona ("pedro","gomez",70);
        Persona persona3= new Persona
("maria","fernandez",25);
        Persona persona4= new Persona ("david","perez",75);
        List<Persona> lista=
Arrays.asList(persona1,persona2,persona3,persona4);
        lista.stream().filter(Predicate.not(Persona::estaJubilado)).forEach(Sy
stem.out::println);

    }
}
```

En este caso habremos realizado una negación de nuestro Predicate y saldrán por la consola

aquellos que no estan jubilados.

```
Persona [nombre=ana, apellidos=sanchez, edad=30]  
Persona [nombre=maria, apellidos=fernandez, edad=25]
```

Otros artículos relacionados

- [Java Stream Reduce , eliminando bucles](#)
- [Java 8 FlatMap y Streams](#)
- [Java Stream Sorted y Comparators](#)
- [Java Predicate](#)