Tabla de Contenidos

- Entity to DTO y constructores
- Entidades y Listas
  - Java 8 constructor Reference Entity to DTO
- Otros artículos relacionados

La conversión de Entity a DTO es una de las conversiones más habituales que tenemos cuando programamos en el día a día en Java. Vamos a ver un par de ejemplos habituales y como podemos usar Java 8 para simplificar la transformación. Supongamos que partimos de 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;
	}
}
```

Esta clase es la que va a realizar las funciones de Entidad. Mientras que vamos a tener otra muy parecida que hará las funciones de DTO.

```
package com.arquitecturajava;

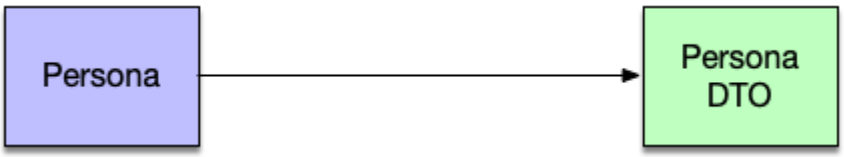
public class PersonaDTO {

	public PersonaDTO(String nombre, String apellidos) {
		super();
		this.nombre = nombre;
		this.apellidos = apellidos;
	}
	private String nombre;
	private String apellidos;
	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 PersonaDTO(Persona persona) {
                super();
                this.nombre= persona.getNombre();
                this.apellidos=persona.getApellidos();
        }
}
```

Entity to DTO y constructores

En este caso las dos clases son muy parecidas y hemos obligado a la clase que es un DTO a recibir como parámetro una Persona a la hora de instanciarla por supuesto puede soportar otros tipos de constructores sin problemas.



La forma más sencilla de utilizarla creando un objeto DTO que recibe como parámetro la Persona.

```
package com.arquitecturajava;

public class Principal {

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

```
                Persona p = new Persona("pepe", "perez", 30);

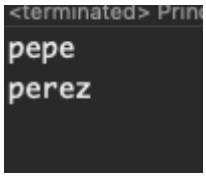
                PersonaDTO dto = new PersonaDTO(p);

                System.out.println(dto.getNombre());
                System.out.println(dto.getApellidos());


        }

}
```

El resultado saldrá por la consola sin ningún problema acabamos de realizar una transformación.



Entidades y Listas

En muchas ocasiones no solo necesitamos convertir una Entidad a un DTO sino que ademas necesitamos hacerlo con una lista de elementos de este tipo de entidades.

```
package com.arquitecturajava;

import java.util.ArrayList;
import java.util.List;

public class Principal2 {

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

```
                Persona p= new Persona ("pepe","perez",30);
                Persona p1 = new Persona("ana", "sanchez", 40);

                List<Persona> lista = new ArrayList<Persona>();
                lista.add(p);
                lista.add(p1);

                List<PersonaDTO> listaDTO = new
ArrayList<PersonaDTO>();
                for (Persona persona : lista) {

                        listaDTO.add(new PersonaDTO(persona));
                }

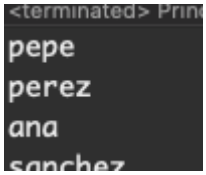
                for (PersonaDTO undto : listaDTO) {

                        System.out.println(undto.getNombre());
                        System.out.println(undto.getApellidos());

                }
        }

}
```

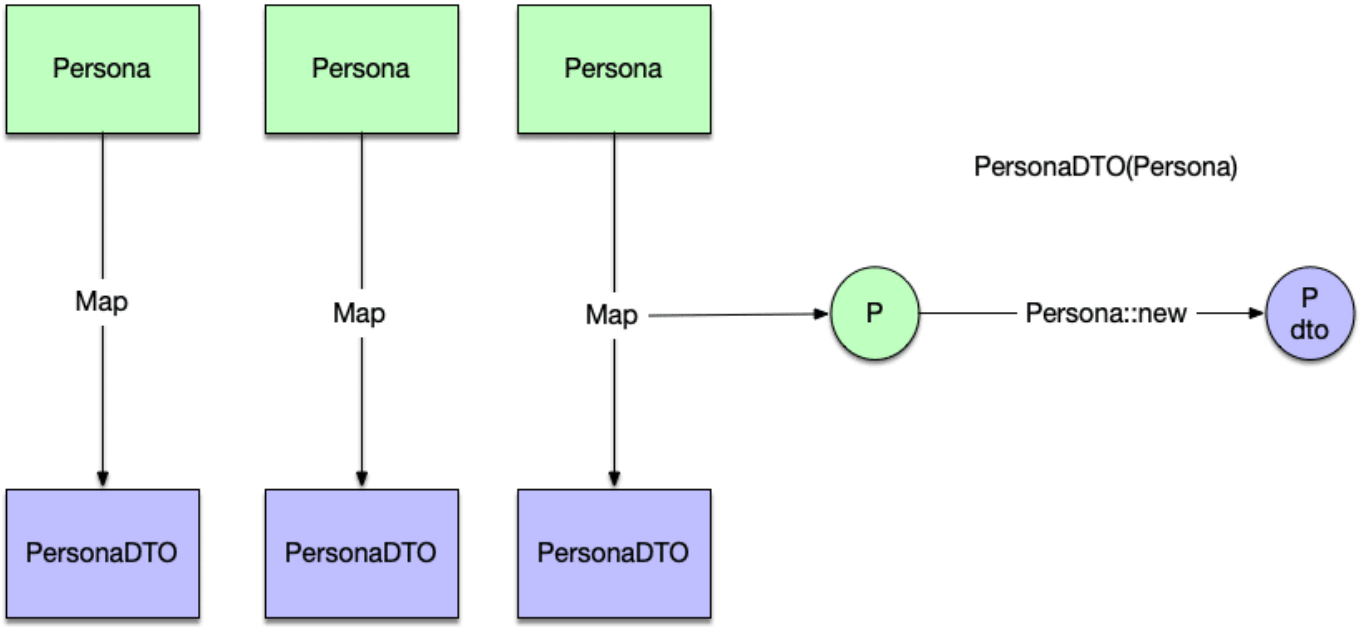
En este caso creamos una lista y añadimos objetos persona a ella . Una vez que tenemos la lista rellena el siguiente paso es utilizar un bucle forEach y generar una nueva lista de DTOs la cual imprimimos por la consola.



Como su puede observar este código es bastante sencillo pero también bastante extenso a la hora de utilizarlo . ¿Como podemos simplificar las cosas para que todo sea más natural con Java 8?

Java 8 constructor Reference Entity to DTO

La solución pasa por hacer uso de una de las capacidades que tiene Java 8 a nivel de métodos de referencia y manejo de constructores.



Veamos su código:

```
package com.arquitecturajava;

import java.util.ArrayList;
import java.util.List;
import java.util.stream.Collectors;

public class Principal3 {

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

```
                Persona p= new Persona ("pepe","perez",30);
                Persona p1 = new Persona("ana", "sanchez", 40);
                List<Persona> lista = new ArrayList<Persona>();
                lista.add(p);
                lista.add(p1);
                List<PersonaDTO>
listaDTO=lista.stream().map(PersonaDTO::new).collect(Collectors.toList
());
                for (PersonaDTO undto : listaDTO) {

                        System.out.println(undto.getNombre());
                        System.out.println(undto.getApellidos());

                }
        }

}
```

El resultado será el mismo pero mucho más compacto y nos ahorraremos el bucle for inicial. Recordemos que los Java constructor reference siempre nos aportan flexibilidad y en este caso los usamos para hacer una transformación de Entity to DTO.

Otros artículos relacionados

- JPA DTO
- Eclipse clases de Dominio
- Java 8 Collectors