JackSon ObjectMapper es uno de los objetos principales de JackSon que es la librería más usada por parte de los desarrolladores Java a la hora hora de manejar información en JSON . Ademas esta librería es la librería estandar en Spring Boot . Vamos a ver un ejemplo sencillo de ObjectMapper y la clase Persona que nos permite leer y escribir un objeto en formato JSON de forma sencilla. Para ello vamos a declarar la clase Persona:

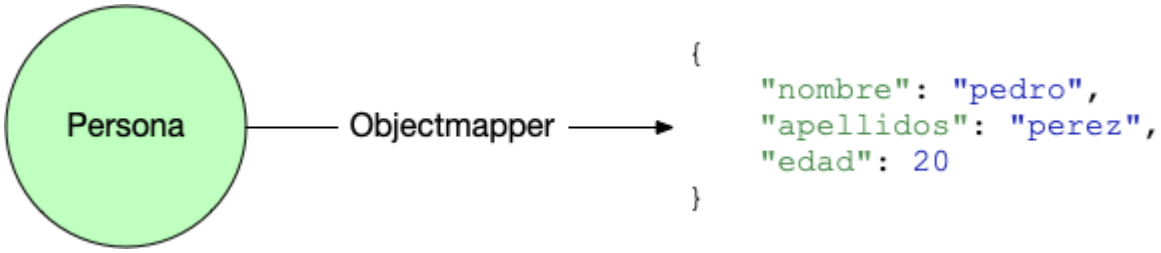
```
package com.arquitecturajava.json1;

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 Persona() {
		super();
	}
	@Override
	public String toString() {
		return "Persona [nombre=" + nombre + ", apellidos=" +
apellidos + ", edad=" + edad + "]";
	}
}
```

Una vez hecho esto nos queda usar el ObjectMapper y su método writeValue para escribir los datos en un fichero JSON.




```
package com.arquitecturajava.json1;

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

import com.fasterxml.jackson.core.exc.StreamWriteException;
import com.fasterxml.jackson.databind.DatabindException;
```

```
import com.fasterxml.jackson.databind.ObjectMapper;

public class Principal {

	public static void main(String[] args) {
		try {
			ObjectMapper mapeador= new ObjectMapper();
			Persona persona= new
Persona("pepe","sanchez",50);
			mapeador.writeValue(new File("persona2.json"),
persona);
		} catch (StreamWriteException e) {
			// TODO Auto-generated catch block
			e.printStackTrace();
		} catch (DatabindException e) {
			// TODO Auto-generated catch block
			e.printStackTrace();
		} catch (IOException e) {
			// TODO Auto-generated catch block
			e.printStackTrace();
		}

	}

}
```

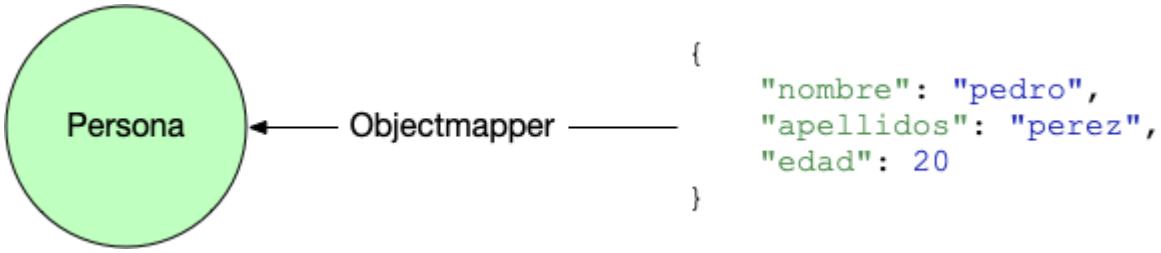
Una vez que tenemos el código lo ejecutamos y el ObjectMapper generará un fichero:

```
{
	"nombre": "pedro",
	"apellidos": "perez",
	"edad": 20
```

```
}
```

Jackson ObjectMapper y escritura

Acabamos de realizar la primera operación que es escribir un objeto a JSON . Vamos a realizar la operación contraria que es leer un JSON y convertirlo en un Objeto.



Veamos el código:

```
public class Principal {

	public static void main(String[] args) {
		try {
			ObjectMapper mapeador= new ObjectMapper();
			Persona persona= mapeador.readValue(new
File("persona.json"),Persona.class);
			System.out.println(persona.getNombre());
			System.out.println(persona.getApellidos());
			System.out.println(persona.getEdad());
		} catch (StreamWriteException e) {
			// TODO Auto-generated catch block
			e.printStackTrace();
		} catch (DatabindException e) {
			// TODO Auto-generated catch block
			e.printStackTrace();
		} catch (IOException e) {
```

```
                        // TODO Auto-generated catch block
                        e.printStackTrace();
                }

        }

}
```

Una vez leido el fichero lo imprimirá por la consola:

```
pedro
perez
20
```

Acabamos de leer un fichero JSON y le hemos convertido en un Objeto. La clase ObjectMapper es muy practica para realizar todo este tipo de funcionalidad:

- POJO to JSON y JSON to POJO
- Servlet JSON y el manejo de Ajax
- Java Properties Files y como usarlos
- File to String Java 8 y manejo de ficheros
- Java Stream File y manejo de ficheros

Otros Enlaces

- Jackson Git