



**INSTITUTO POLITÉCNICO NACIONAL  
ESCUELA SUPERIOR DE CÓMPUTO**

**Desarrollo de Sistemas Distribuidos**



# Actividad 17

## JSON

**PROFESOR:** Pineda Guerrero Carlos

**ALUMNA:** Karina

**GRUPO:** 4CV11

## 1. Compilar y ejecutar el programa [EjemploGSON.java](#)

```
kary@DESKTOP-I4R5UOI: /mnt/ x + v
kary@DESKTOP-I4R5UOI: /mnt/c/Users/Karina/Desktop/SISDIS/17.JSON$ javac -cp gson-2.8.6.jar EjemploGSON.java
kary@DESKTOP-I4R5UOI: /mnt/c/Users/Karina/Desktop/SISDIS/17.JSON$ java -cp gson-2.8.6.jar:. EjemploGSON
[{"nombre": "Hugo", "edad": 20, "sueldo": 1000.0, "fecha_ingreso": "2020-01-01T20:10:00.000"}, {"nombre": "Paco", "edad": 21, "sueldo": 2000.0, "fecha_ingreso": "2019-10-01T10:15:00.000"}, {"nombre": "Luis", "edad": 22, "sueldo": 3000.0, "fecha_ingreso": "2018-11-01T00:00:00.000"}]
Hugo 20 1000.0 2020-01-01 20:10:00.0
Paco 21 2000.0 2019-10-01 10:15:00.0
Luis 22 3000.0 2018-11-01 00:00:00.0
kary@DESKTOP-I4R5UOI: /mnt/c/Users/Karina/Desktop/SISDIS/17.JSON$
```

2. Agregar un nuevo campo a la clase Empleado, el campo "jefe" de tipo Empleado.
3. Modificar el constructor de la clase Empleado para incluir la inicialización del campo "jefe", tal como se muestra:

```
static class Empleado
{
    String nombre;
    int edad;
    float sueldo;
    Timestamp fecha_ingreso;
    Empleado jefe;
    Empleado(String nombre, int edad, float sueldo, Timestamp fecha_ingreso, Empleado jefe)
    {
        this.nombre = nombre;
        this.edad = edad;
        this.sueldo = sueldo;
        this.fecha_ingreso = fecha_ingreso;
        this.jefe = jefe != null ? jefe : this;
    }
}
```

4. Crear los tres empleados de manera que Hugo sea jefe de si mismo, Hugo sea jefe de Paco, y Paco sea jefe de Luis, tal como se muestra:

```
Empleado[] e = new Empleado[3];
e[0] = new Empleado("Hugo", 20, 1000, Timestamp.valueOf("2020-01-01 20:10:00"), null);
e[1] = new Empleado("Paco", 21, 2000, Timestamp.valueOf("2019-10-01 10:15:00"), e[0]);
e[2] = new Empleado("Luis", 22, 3000, Timestamp.valueOf("2018-11-01 00:00:00"), e[1]);
```

5. Desplegar los datos de cada empleado:

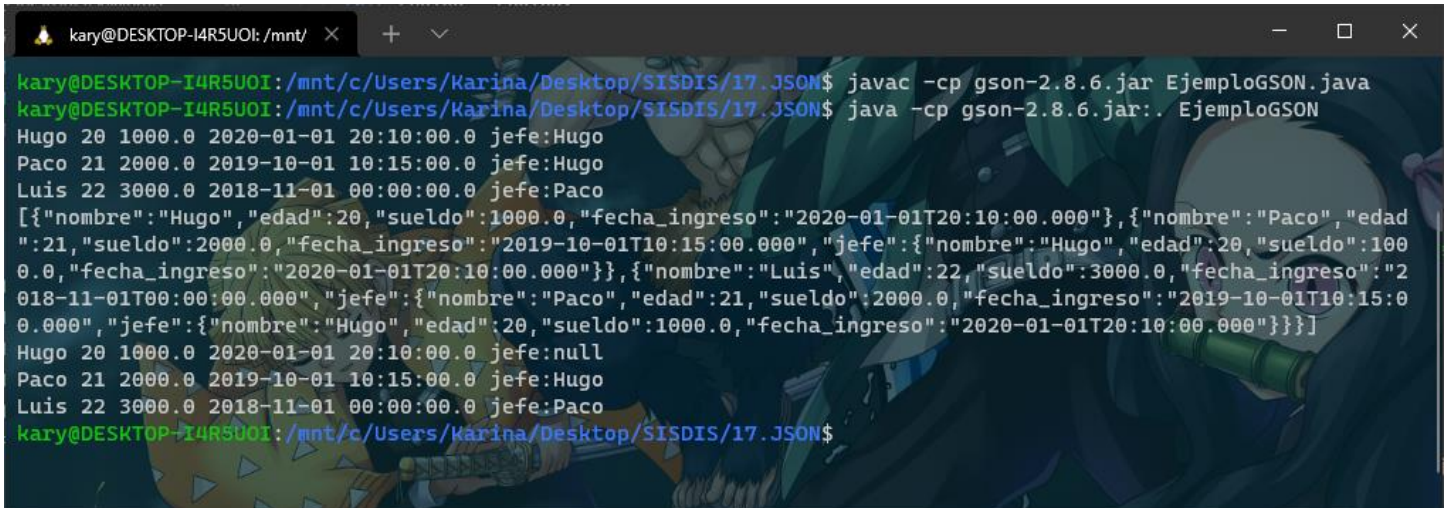
```
for (int i = 0; i < e.length; i++)
    System.out.println(e[i].nombre + " " + e[i].edad + " " + e[i].sueldo + " " +
e[i].fecha_ingreso + " jefe:" + e[i].jefe.nombre);
```

6. Cambiar el despliegue del arreglo deserializado, de la siguiente manera:

```
for (int i = 0; i < v.length; i++)
```

```
System.out.println(v[i].nombre + " " + v[i].edad + " " + v[i].sueldo + " " +  
v[i].fecha_ingreso + " jefe:" + (v[i].jefe != null ? v[i].jefe.nombre : null));
```

## 7. Compilar y ejecutar el programa.



```
kary@DESKTOP-I4R5UOI: /mnt/ x + v  
kary@DESKTOP-I4R5UOI: /mnt/c/Users/Karina/Desktop/SISDIS/17.JSON$ javac -cp gson-2.8.6.jar EjemploGSON.java  
kary@DESKTOP-I4R5UOI: /mnt/c/Users/Karina/Desktop/SISDIS/17.JSON$ java -cp gson-2.8.6.jar:. EjemploGSON  
Hugo 20 1000.0 2020-01-01 20:10:00.0 jefe:Hugo  
Paco 21 2000.0 2019-10-01 10:15:00.0 jefe:Hugo  
Luis 22 3000.0 2018-11-01 00:00:00.0 jefe:Paco  
[{"nombre":"Hugo","edad":20,"sueldo":1000.0,"fecha_ingreso":"2020-01-01T20:10:00.000"}, {"nombre":"Paco","edad":21,"sueldo":2000.0,"fecha_ingreso":"2019-10-01T10:15:00.000"}, {"nombre":"Luis","edad":22,"sueldo":3000.0,"fecha_ingreso":"2018-11-01T00:00:00.000"}, {"nombre":"Hugo","edad":20,"sueldo":1000.0,"fecha_ingreso":"2020-01-01T20:10:00.000"}]  
Hugo 20 1000.0 2020-01-01 20:10:00.0 jefe:null  
Paco 21 2000.0 2019-10-01 10:15:00.0 jefe:Hugo  
Luis 22 3000.0 2018-11-01 00:00:00.0 jefe:Paco  
kary@DESKTOP-I4R5UOI: /mnt/c/Users/Karina/Desktop/SISDIS/17.JSON$
```

7.1 ¿Qué despliega la serialización?

7.2 Al serializar el empleado Hugo ¿Se muestra la auto-referencia que hace el empleado Hugo a sí mismo?

7.3 ¿Hay alguna redundancia en los objetos serializados?

7.4 ¿Qué despliega la deserialización?

7.5 ¿Qué concluye sobre GSON y la forma en que serializa y des-serializa las referencias a objetos?