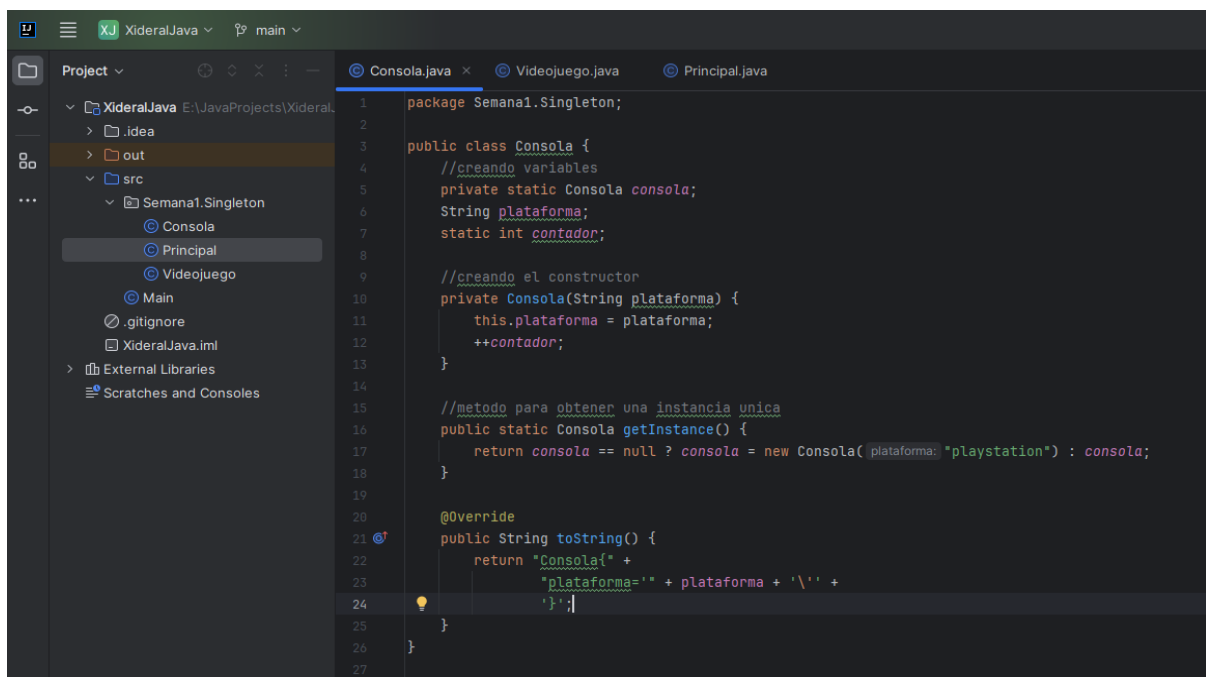


SINGLETON JAVA

Creo un Java Project nuevo donde se llama XideralJava, donde creo 3 clases, una llamada Consola, otra llamada Videojuego y mi clase Principal.

CLASE CONSOLA

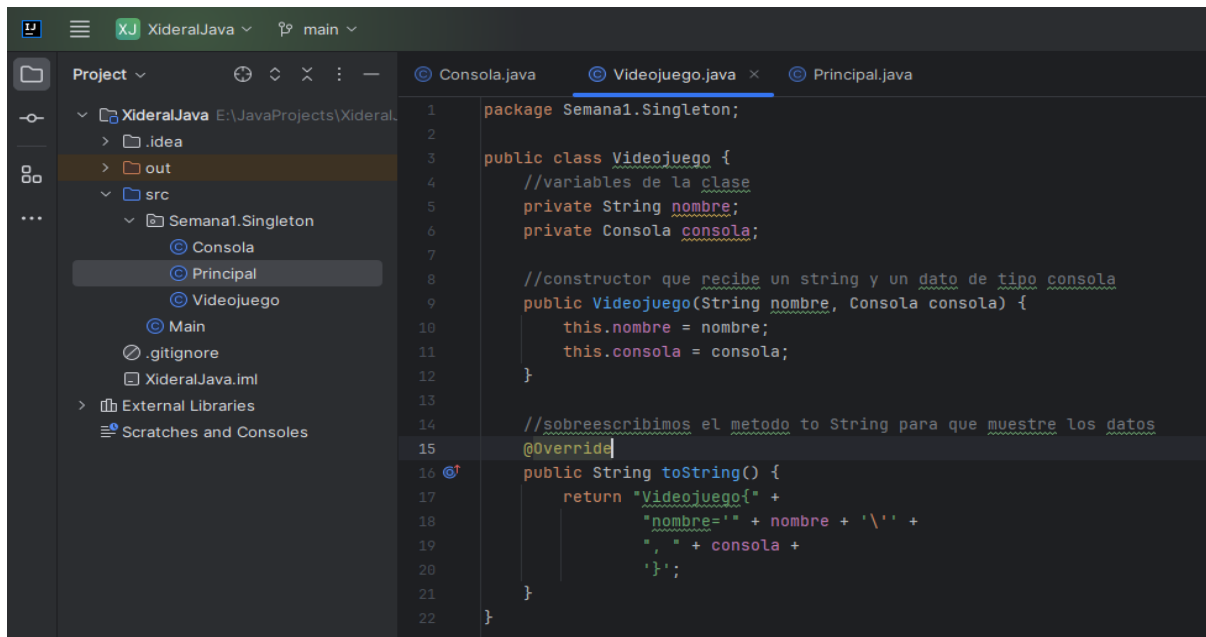
La clase consola será nuestro singleton, la cual será nuestra única instancia, en el metodo getInstance(), se utilizó un operador ternario que se utiliza para validar si consola es null, si es null entonces creamos un objeto de tipo consola, pero si es falso no crea ningún objeto.



```
1 package Semanal.Singleton;
2
3 public class Consola {
4     //creando variables
5     private static Consola consola;
6     String plataforma;
7     static int contador;
8
9     //creando el constructor
10    private Consola(String plataforma) {
11        this.plataforma = plataforma;
12        ++contador;
13    }
14
15    //metodo para obtener una instancia unica
16    public static Consola getInstance() {
17        return consola == null ? consola = new Consola( plataforma: "playstation") : consola;
18    }
19
20    @Override
21    public String toString() {
22        return "Consola{" +
23            "plataforma=" + plataforma + '\'' +
24            '}';
25    }
26 }
27
```

CLASE VIDEOJUEGO

Esta clase es como otras clases, en la cual su constructor tendrá como parámetros un valor de tipo String y otro de tipo Consola.

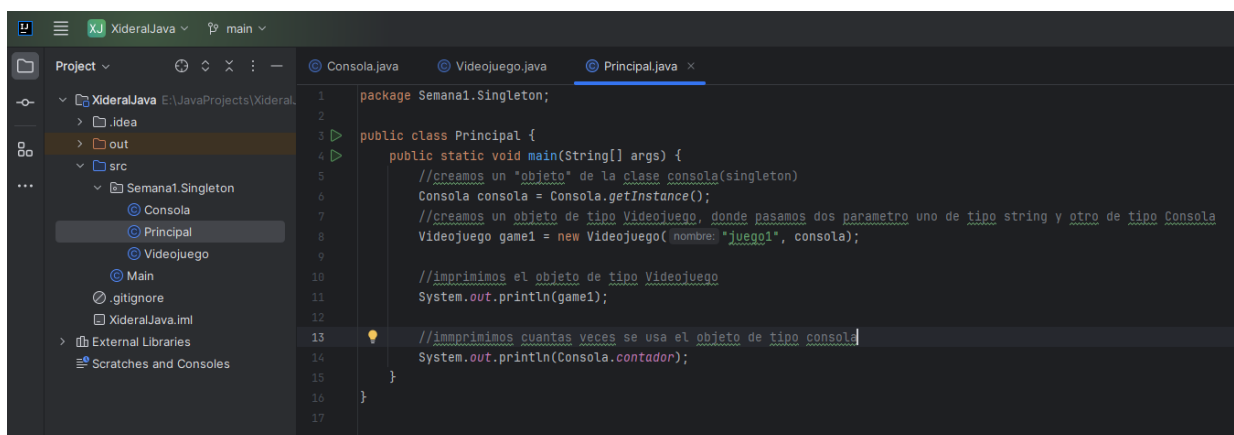


The screenshot shows an IDE with the 'Videojuego.java' file open. The code defines a class 'Videojuego' within the 'Semana1.Singleton' package. It has two private attributes: 'nombre' of type String and 'consola' of type Consola. The constructor 'Videojuego(String nombre, Consola consola)' initializes these attributes. An '@Override' annotation is placed above the 'toString()' method, which returns a string representation of the object, including the name and console.

```
1 package Semana1.Singleton;
2
3 public class Videojuego {
4     //variables de la clase
5     private String nombre;
6     private Consola consola;
7
8     //constructor que recibe un string y un dato de tipo consola
9     public Videojuego(String nombre, Consola consola) {
10         this.nombre = nombre;
11         this.consola = consola;
12     }
13
14     //sobreescribimos el metodo to String para que muestre los datos
15     @Override
16     public String toString() {
17         return "Videojuego{" +
18             "nombre='" + nombre + '\'' +
19             ", " + consola +
20             '}' ;
21     }
22 }
```

CLASE PRINCIPAL

En la clase principal estará nuestro main, y aquí es donde crearemos nuestros objetos, pero como nuestra clase Consola es nuestro singleton solo realizará una instancia y si creamos más objetos de tipo Consola solo nos dará como resultado que se creó 1.



The screenshot shows the 'Principal.java' file in the IDE. It contains the 'main' method of the 'Principal' class. The code creates a 'Consola' singleton instance using 'Consola.getInstance()'. Then, it creates a 'Videojuego' object named 'game1' by passing the singleton console and the string 'juego1'. Finally, it prints the 'game1' object and the 'contador' attribute of the 'Consola' singleton to the console.

```
1 package Semana1.Singleton;
2
3 public class Principal {
4     public static void main(String[] args) {
5         //creamos un "objeto" de la clase consola(singleton)
6         Consola consola = Consola.getInstance();
7         //creamos un objeto de tipo Videojuego, donde pasamos dos parametro uno de tipo string y otro de tipo Consola
8         Videojuego game1 = new Videojuego(nombre: "juego1", consola);
9
10        //imprimos el objeto de tipo Videojuego
11        System.out.println(game1);
12
13        //imprimos cuantas veces se usa el objeto de tipo consola
14        System.out.println(Consola.contador);
15    }
16 }
17 }
```