cedia | TMO Training Management Office

info@cedia.org.ec
cedia.edu.ec

CURSO DE JAVASCRIPT

Corporación Ecuatoriana para el Desarrollo de la Investigación y la Academia

Ing. María Fernanda Granda J., PhD

25-11-2025

"Conectando ideas, transformando sociedades"

Módulo 3: Programación Orientada a Objetos

- Instalar Nodejs
- Polimorfismo

Instalar Node js

- Bajarse el instalador desde https://nodejs.org/es/download
- Ejecutarlo para instalarlo
- Configurar en Este Equipo/Propiedades/Variables de Entorno
- Agregar una variable para Nodejs con el valor de la ubicación C:\Program Files\nodejs
- Ya se puede probar el código en la terminal de Visual Studio Code

Polimorfismo

```
class Instrumento {
    constructor(nombre){
        // Propiedad o característica
        this.nombre = nombre
    }

    // Método
    tocar(){
        console.log(`${this.nombre} está produciendo un sonido`)
    }
}
```

Polimorfismo

```
class Guitarra extends Instrumento{
    constructor(){
        super('Guitarra')
    }

    // POLIFORMISMO: es la capacidad de múltiples clases a responder distinto a una misma función o método

    // Sobreescribir o pisar [OVERRIDE] el método heredado
    tocar(){
        console.log("\x1b[31m%s\x1b[0m", `¡La ${this.nombre} está haciendo un solo impresionante!`)
    }
}
```

Colores Ansi usando caracteres de Escape

```
console.log("\x1b[31m%s\x1b[0m", "Este texto es rojo");
console.log("\x1b[32m%s\x1b[0m", "Este texto es verde");
console.log("\x1b[33m%s\x1b[0m", "Este texto es amarillo");
console.log("\x1b[34m%s\x1b[0m", "Este texto es azul");
console.log("\x1b[35m%s\x1b[0m", "Este texto es magenta");
console.log("\x1b[36m%s\x1b[0m", "Este texto es cian");
```

Polimorfismo

```
class Batería extends Instrumento{
    constructor(){
        super('Batería')
    }

    // POLIFORMISMO: es la capacidad de múltiples clases a responder distinto a una misma función o método

    // Sobreescribir o pisar [OVERRIDE] el método heredado
    tocar(){
        console.log("\x1b[32m%s\x1b[0m", `¡La ${this.nombre} está tocando los tambores a un gran ritmo!`)
    }
}
```

Polimorfismo

```
class Piano extends Instrumento{
    constructor(){
        super('Piano')
    }

    // POLIFORMISMO: es la capacidad de múltiples clases a responder distinto a una misma función o método

    // Sobreescribir o pisar [OVERRIDE] el método heredado
    tocar(){
        console.log("\x1b[34m%s\x1b[0m", `¡El ${this.nombre} está haciendo unas hermosas notas pentatónicas de blues!`)
    }
}
```

Polimorfismo

```
const guitarra = new Guitarra()
const bateria = new Bateria()
const piano = new Piano()


guitarra.tocar()
bateria.tocar()
piano.tocar()
```

Polimorfismo

```
function showBanda(instrumento){
    instrumento.tocar()
}

const guitarra = new Guitarra()
const bateria = new Bateria()
const piano = new Piano()


showBanda(guitarra)
showBanda(bateria)
showBanda(piano)
```

Probar el código

- ¿Qué pasa si agregamos un nuevo instrumento?.
- Agregar un instrumento y comprobar que siga funcionando el polimorfimo.
- Subir la tarea en el entorno virtual del Curso.

cedia | TMO Training Management Office

GRACIAS

cedia.edu.ec
info@cedia.org.ec
(+593) 7-407-9300