DigitalHouse>

Certified Tech Developer
The Ultimate Degree

Front End III

Creando una clase en JavaScript

Objetivo

Veamos qué sucede internamente cuando creamos una clase en JavaScript.

Supongamos que creamos una clase muy simple en JavaScript llamada Pelicula:📌

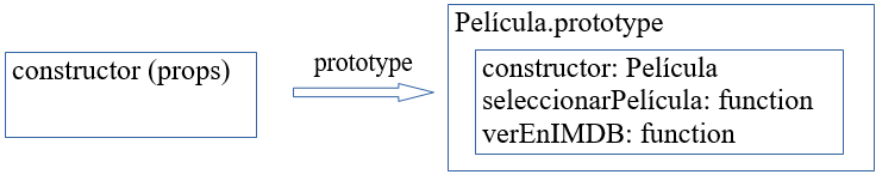
```
class Pelicula {

      constructor(props) { }

      seleccionarPelicula() { }

      verEnIMDB() { }

}
```

Internamente, lo que class **Pelicula {...}** hace es:

1. Crear una función llamada **Pelicula**, que es el resultado de la declaración de la clase. El código de esta función se toma del constructor (se supone un código vacío si no escribimos un constructor).

2. Guardar las funciones de clase, como **seleccionarPelicula** y **verEnIMDB**, en **Pelicula.prototype**.

3. El constructor de prototype apunta a la función **Pelicula**.



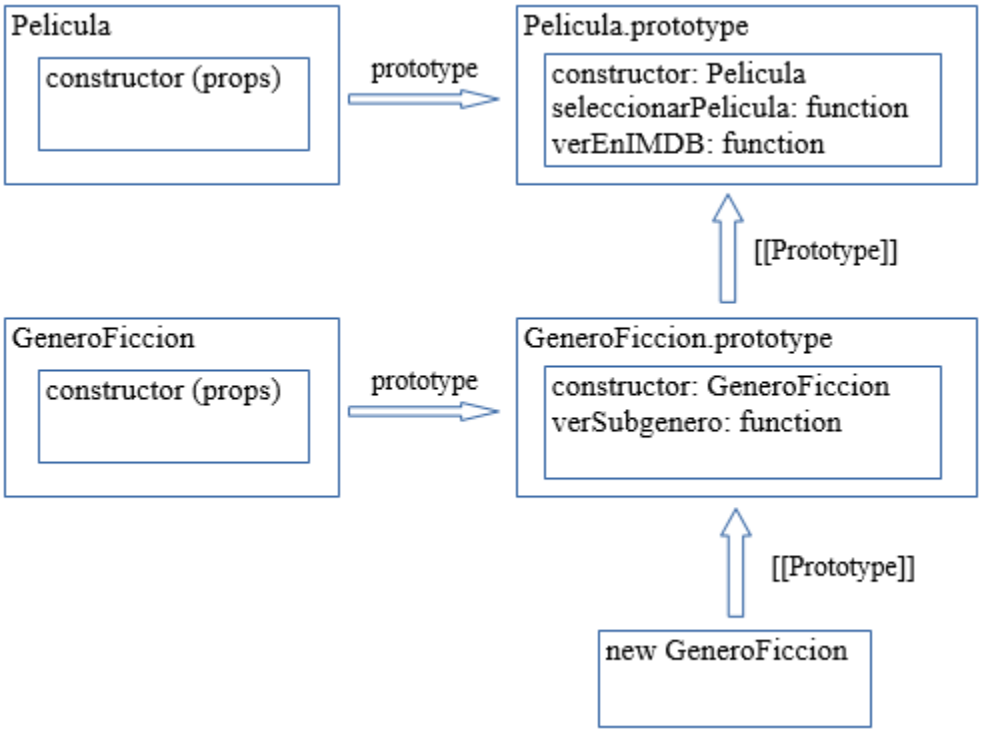
Ahora veamos qué sucede cuando creamos una jerarquía de clases en JavaScript:📌

Como ejemplo, creemos una subclase llamada **GeneroFiccion**:

```
class GeneroFiccion extends Pelicula {

        constructor(props) {

        super(props);

        }

        verSubgenero() { }

}
```

Internamente, **extends** trabaja utilizando la mecánica de prototipos. Apunta **GeneroFiccion.prototype.[[prototype]]** a **Pelicula.prototype**. De esta manera, si no se encuentra una función en **GeneroFiccion.prototype**, JavaScript lo toma de **Pelicula.prototype**.

Este es el diagrama de clases resultante:



Ahora, aunque la única función que tiene la clase **GeneroFicción** es **verSubgenero**, si creamos un objeto de clase **GeneroFiccion** con new **GeneroFiccion()** y llamamos a la función **seleccionarPelicula**, esta estará disponible a través de la cadena de prototipos. Esto permite la reutilización al especializar con subclases en JavaScript.

¡Hasta la próxima!