**Prototype en JavaScript:** Es un objeto del cual un objeto hereda propiedades y métodos. Cada objeto en JavaScript tiene un prototipo. Cuando creas un objeto en JavaScript, este tiene un enlace oculto a otro objeto llamado "prototype".

**¿Cómo Funciona el Prototipo?** La **cadena de prototipos** es el camino que JavaScript sigue para buscar propiedades y métodos en los objetos. Cada objeto tiene un enlace oculto llamado \_\_proto\_\_ que apunta a su prototype. Este prototype es otro objeto que actúa como "base" del objeto original. Si intentas acceder a una propiedad o método en un objeto y este no la tiene, JavaScript busca esa propiedad o método en su prototype. Si tampoco está allí, sigue buscando en el prototype del prototype, y así sucesivamente, hasta llegar a null.

**Herencia de prototipos**: Permite crear relaciones entre objetos, de modo que los objetos más específicos puedan usar las características de objetos más generales.

**Propiedad Prototype en Funciones Constructoras:** Las funciones constructoras tienen una propiedad especial llamada **prototype**. Esta propiedad se utiliza para añadir métodos y propiedades que serán compartidos por todas las instancias creadas con esa función constructora.

**Métodos Estáticos y prototype:** Los métodos estáticos definidos en una función constructora no se heredan por las instancias del objeto. Estos métodos están asociados directamente con la función constructora en sí, no con el prototipo.

Texto

Descripción generada automáticamente

**Propiedades y Métodos que Acceden al Prototype**

* **Object.getPrototypeOf(obj):** Devuelve el prototipo del objeto obj
* **Object.setPrototypeOf(obj, prototype)**: Establece el prototipo del objeto obj a prototype.

Todos los métodos que pertenecen a Object.prototype están disponibles para todos los objetos. Como por ejemplo, toString(), hasOwnProperty(), valueOf(), etc. Estos métodos están disponibles para todos los objetos, ya que todos los objetos en JavaScript heredan de Object.prototype.

**Propiedades No Accesibles a través del Prototipo:** Propiedades propias del objeto: Si una propiedad o método está directamente definido en un objeto y no en su prototipo, no será accesible a través del prototipo.