Prototipos – Funciones Constructor

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Object.create()**  const herosMethos = {  saludar: function(){  console.log(`Soy superheroe! ${this.nombre}`)  }  }  function Heros(name){  const hero = Object.create(herosMethos)  hero.nombre = name  return hero  }  const zelda = Heros('Zelda')  zelda.saludar()     |  | | --- | | Creamos “instancias” de hero con la función Hero pasandole el nombre como argumento para usar en el atributo correspondiente, en este caso ***hero.nombre,*** que recibe ***name*** |  |  | | --- | | function Heros(name){  const hero = Object.create(**Heros.*prototype***)  hero.nombre = name  return hero  }  Heros.***prototype*.**saludar = function(){  console.log(`Soy heroe! ${this.nombre}`)  } |  |  | | --- | | La función Heros sirve como ***constructor***, se crea una instancia del objeto herosMethos con Object.create en la constante ***hero.*** Después de crear el objeto hero le agregamos el **atributo** nombre con ***hero.nombre.*** Es muy importante ***return*** “hero”, para que devuelva el objeto creado cuando se hace la instancia |  |  | | --- | | Object.create(objeto) puede **crear instancias de objetos sin crear antes constructores.** Con este método puede crear un nuevo objeto basado en cualquier otro objeto existente. | | En ES5 se introdujo una mejora sintáctica (syntactic sugar) con la keyword ***new*** y el uso de ***this*** para hacer referencia al objeto del contexto actual.  function Heros(name){  **// El keyword new en la instanciación crea el "prototype" y la referencia "this" en el constructor que sería esta función, entonces la línea comentada de abajo es reemplazada**  // const hero = Object.create(Heros.prototype)  // this = Object.create(Heros.prototype)  **// Al crearse la referencia "this" ya no hace falta hero.propiedad**  **// hero.nombre = name**  ***this***.nombre = name  **// "return" ocurre implícitamente, ya no hey que incluirlo**  // return this  // **Incluir los métodos en el constructor crea la función en todas las instancias haciendolo ineficiente. Agregarlo al "prototype" hace instancias mas livianas que pueden usar el método buscando en la cadena prototipal**  // this.saludar = function(){console.log(`Soy heroe! ${this.nombre}`)}  }  ***Heros.prototype.saludar*** = function(){  console.log(`Soy superheroe con "new"! ${this.nombre} `)  }    const zelda = ***new*** Heros('Zelda')  zelda.saludar()  const link = ***new*** Heros('Link')  link.saludar() |