FMO

Técnicas de Programación para Internet

Ing. Diego Herrera

Agenda

Herencia en JavaScript

Prototype

Factory function

Crear un repositorio en Github

Clonar repositorio

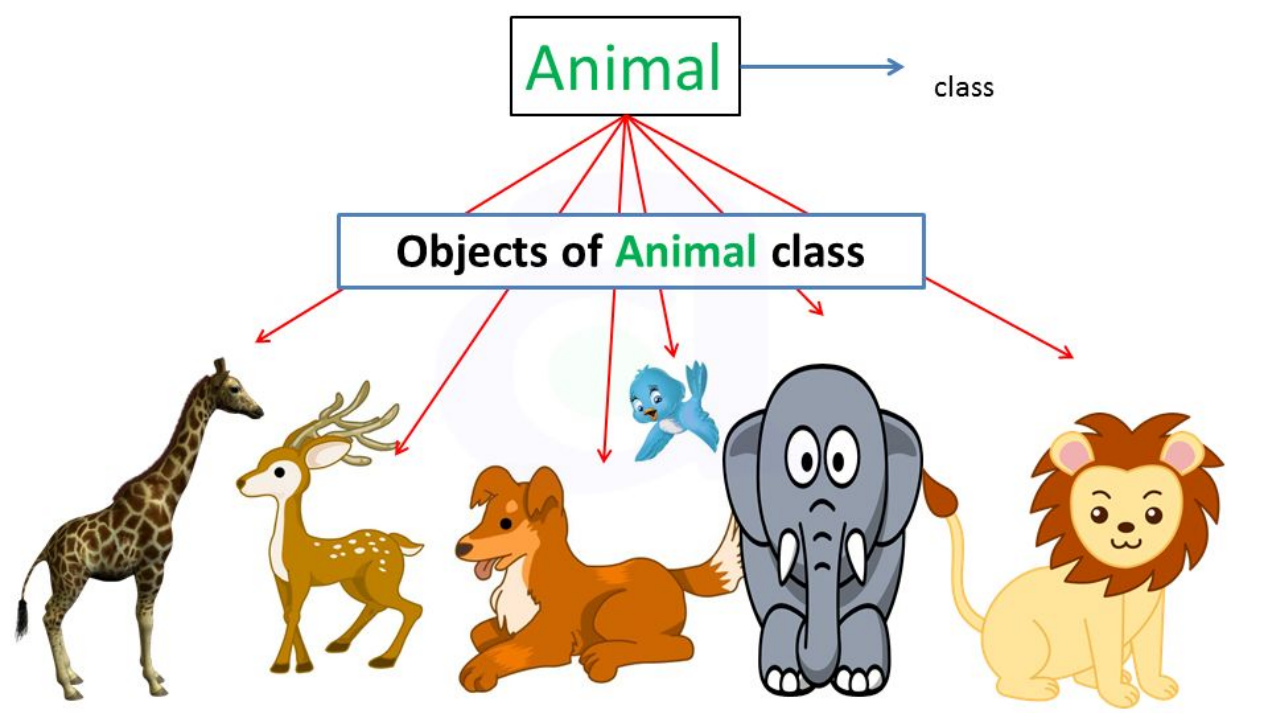
Modificar repositorio local y sincronizar con repositorio remoto

Herencia en JavaScript

La herencia en POO es exactamente como sucede en la vida real, nosotros tenemos ancestros de los cuales heredamos algunos rasgos e incluso comportamientos, puede que heredemos el color de piel de nuestro padre y el color de ojos de nuestra madre, por ejemplo.

En Javascript sucede algo similar, nuestros objetos pueden transferir información y compartir propiedades mediante la herencia.

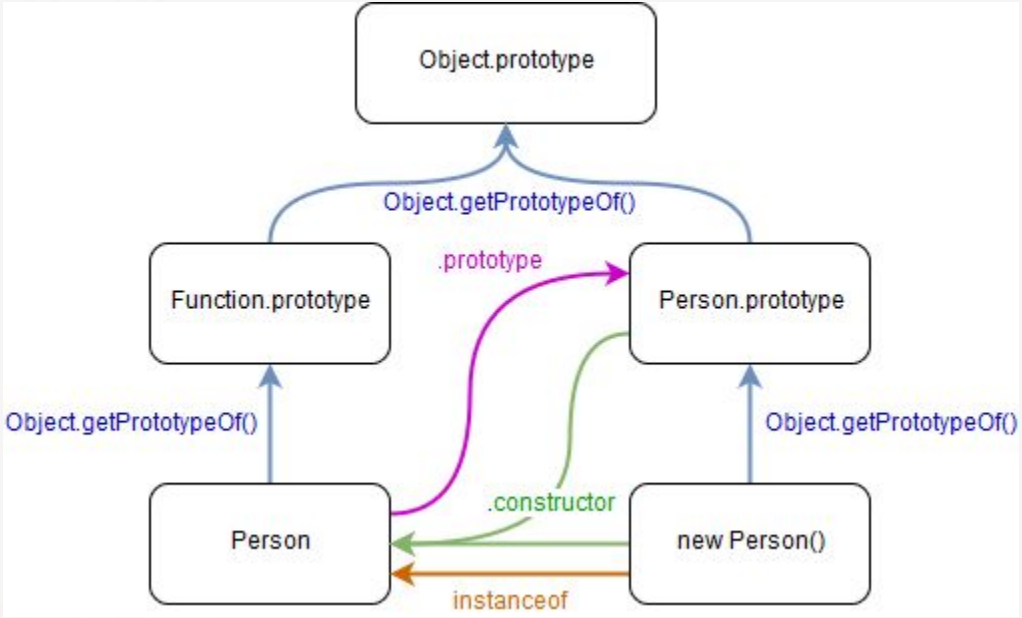
Herencia en JavaScript



Prototype

El prototype es un mecanismo mediante el cual los objetos en JavaScript heredan características entre sí. Javascript a menudo es llamado lenguaje basado en prototipos.

Prototype


Object.prototype
Object.getPrototypeOf()
.prototype
Function.prototype
Person.prototype
Object.getPrototypeOf()
Object.getPrototypeOf()
.constructor
Person
new Person()
instanceof

Prototype

```
1  function Persona(nombre, apellido) {
2      this.nombre = nombre;
3      this.apellido = apellido;
4      this.saludar = function() {
5          return "Hola, mundo!";
6      }
7  }
8
9  const diego = new Persona("Diego", "Herrera");
10
11 console.log(Persona.prototype);
```

Propiedad Prototype

```
13  Persona.prototype.despedida = function() {
14      return "Adiós";
15  }
16
17  console.log(Persona.prototype);
18  console.log(Object.prototype)
19  console.log(diego instanceof Persona);
20
```

Propiedad Prototype

```
13  Persona.prototype.despedida = function() {
14      return "Adiós";
15  }
16
17  console.log(Persona.prototype);
18  console.log(Object.prototype)
19  console.log(diego instanceof Persona);
20
```

Propiedad Prototype

```
21  function Estudiante(nombre, apellido) {
22      this.nombre = nombre;
23      this.apellido = apellido;
24  }
25
26  Estudiante.prototype = Persona.prototype
27
28  Estudiante.prototype.despedida = function() {
29      return "Nos vemos!";
30  } // Prueba imprimir el método despedida tanto en persona como en entudiante
31
32  let miPersona = new Persona("Jorge", "Ramírez");
```

Object.create

```
1  function Pokemon(nombre, tipo) {
2      this.nombre = nombre;
3      this.tipo = tipo;
4  }
5
6  function TipoFuego(nombre, apellido) {
7      Pokemon.call(this, nombre, apellido);
8  }
9
10 TipoFuego.prototype = Object.create(Pokemon.prototype);
11 TipoFuego.prototype.constructor = TipoFuego;
12
13 console.log(TipoFuego.prototype);
14
15 const charizard = new TipoFuego("Charizard", "fuego");
16
17 console.log(charizard);
```

Ejercicio

Modifica el código visto en clase, y crea un archivo llamado pokedex.js y dentro crea un objeto Pokemon cuyo constructor se inicialice con el nombre del pokemon, su nickname, y el tipo. Dentro asigna esas propiedades. Create otros 3 objetos de tipos diferentes y lista sus debilidades.

Luego create una clase de un pokemon específico de cada tipo y que este tenga la cadena de herencia antes mencionada y lista los ataques disponibles de los 3 pokemon.

Crea un repositorio en Github que se llame pokedex y sincronizalo que tu trabajo realizado. Envía tu enlace de repositorio en el apartado de entrega que estará disponible.