Prototypal Object Creation Patterns in Javascript (ECMAScript 6+)

v. 3

Pattern | Code | Object Created

Constructor

1a

```
function Dog (name, breed) {
  this.name = name;
  this.breed = breed;
  return this;
}
```

1b

```
Dog.prototype.jump = function() {
  console.log(`${this.name} jumps!`);
}
```

Dog

```
{
  name: 'Dog',
  length: 2,
  prototype: { jump: jump, constructor: Dog, __proto__: Object.prototype },
  __proto__: Function.prototype,
  constructor: Function
}
```

Pseudo-Classical

2a

```
let rex = new Dog('Rex', 'Poodle');
rex.breed;  // 'Poodle'
```

2b

```
rex.jump(); // logs 'Rex jumps!'
```

rex

```
{
  name:  'Rex',
  breed:  'Poodle',
  __proto__: Dog.prototype,
  constructor: Dog
}
```

Prototype

3

```
let fido = Object.create(Dog.prototype);
fido.jump(); // logs 'undefined jumps!'
fido.breed;  // undefined
```

fido

```
{
  __proto__: Dog.prototype,
  constructor: Dog
}
```

Objects Linking to Other Objects (OLOO)

4

```
let rexBaby = Object.create(rex);
rexBaby.jump(); // logs 'Rex jumps!'
rexBaby.name;  // 'Rex'
```

rexBaby

```
{
  __proto__: rex,
  constructor: Dog
}
```

5

```
let Cat = {
    init: function(name) {
        this.name = name;
        return this;
    },
    speak: function() { console.log(
        `${this.name} meows`);
    }
};
let garfield = Object.create(Cat).init('Garfield');
garfield.speak(); // logs 'Garfield meows'
```

garfield

```
{
  name: 'Garfield',
  __proto__: Cat,
  constructor: Object
}
```

Legend

Property
Native
Own

typeof Value
'string'
'object'
'function'

Source: Vahid Dejwakh, 2021