

JS - Inheritance

Monday, May 1, 2023 11:21 AM

Inheritance with classes

ES6 je uveo class sintaksu, ali u pozadini se koristi nasljeđivanje pomoću objekta (prototypal inheritance)
Kada se koristi ključna riječ class JS u pozadini kreira funkciju

1. Kreiraj klasu Osoba

```
> class Osoba {  
  govor() {  
    return 'Govorim...';  
  }  
};
```

2. Kreiraj objekte ivica i marica

```
> const ivica = new Osoba();  
> const marica = new Osoba();
```

3. Pozovi funkciju govor na oba objekta

```
> ivica.govor();  
< 'Govorim...'  
  
> marica.govor();  
< 'Govorim...'
```

4. Ispiši objekt ivica

```
> ivica  
< Osoba {} --> ivica je objekt tipa Osoba, nema funkcije govor
```

Ako prikažemo cijelo stablo ispod objekta Osoba



funkcija govor se nalazi na prototype objektu na osnovu kojeg je kreiran objekt ivica

5. Usporedba objekta ivica i klase Osoba

```

> ivica
< ▼ Osoba {} ⓘ
  ▾ [[Prototype]]: Object
    ▶ constructor: class Osoba
    ▶ govor: f govor()
  ▾ [[Prototype]]: Object
    ▶ constructor: f Object()
    ▶ hasOwnProperty: f hasOwnProperty()
    ▶ isPrototypeOf: f isPrototypeOf()
    ▶ propertyIsEnumerable: f propertyIsEnumerable()
    ▶ toLocaleString: f toLocaleString()
    ▶ toString: f toString()
    ▶ valueOf: f valueOf()
    ▶ __defineGetter__: f __defineGetter__()
    ▶ __defineSetter__: f __defineSetter__()
    ▶ __lookupGetter__: f __lookupGetter__()
    ▶ __lookupSetter__: f __lookupSetter__()
    ▶ __proto__: (...)
    ▶ get __proto__: f __proto__()
    ▶ set __proto__: f __proto__()
> ivica.__proto__
< ▼ {constructor: f, govor: f} ⓘ
  ▶ constructor: class Osoba
  ▶ govor: f govor()
  ▶ [[Prototype]]: Object
> Osoba.prototype
< ▼ {constructor: f, govor: f} ⓘ
  ▶ constructor: class Osoba
  ▶ govor: f govor()
  ▶ [[Prototype]]: Object
> Osoba.prototype === ivica.__proto__
< true

```

6. Ažuriranje funkcije govor klase Osoba

```

> Osoba.prototype.govor = function() {
  return 'Novi i poboljsani govor...';
};
< f () {
  return 'Novi i poboljsani govor...';
}
> ivica.govor();
< 'Novi i poboljsani govor...'
> |

```

7. Primjer nasljeđivanja

```

> class Zivotinja {
  hodanje() {
    return 'Ja hodam...';
  }
}
< undefined
> const pas = new Zivotinja();
< undefined
> pas.hodanje();
< 'Ja hodam...'
> class Ptica extends Zivotinja {
  letenje() {
    return 'Ja letim...';
  }
}
< undefined
> const kos = new Ptica();
< undefined
> kos.letenje();
< 'Ja letim...'
> kos.hodanje();
< 'Ja hodam...'
> pas.letenje();
✖ ▶ Uncaught TypeError: pas.letenje is not a function
  at <anonymous>:1:5
> -

```

Inheritance with objects (prototypes)

1. Kreiranje objekta Osoba pomoću funkcije i na osnovu njega objekta ivica - ovakva funkcija se naziva konstruktor funkcija

Opcija 1 - funkcija govor je definirana na prototipu objekta Osoba i smatra se metodom

```
> function Osoba() {}
< undefined

> Osoba.prototype.govor = function() {
  return 'Govorim...';
};
< f () {
  return 'Govorim...';
}

> const ivica = new Osoba();
< undefined

> ivica.govor();
< 'Govorim...'

> ivica
< ▼ Osoba {} ⓘ
  ▾ [[Prototype]]: Object
    ► govor: f ()
    ► constructor: f Osoba()
    ► [[Prototype]]: Object
>
```

Opcija 2 - ovdje je funkcija govor direktno na objektu Osoba a ne na njegovom prototipu, i smatra se svojstvom objekta Osoba, nalazit će se na svakoj instanci

```
> function Osoba() {
  this.govor = function() {
    return 'Govorim...';
  }
}
< undefined

> const ivica = new Osoba();
< undefined

> ivica.govor();
< 'Govorim...'

> ivica
< ▼ Osoba {govor: f} ⓘ
  ► govor: f ()
  ▾ [[Prototype]]: Object
    ► constructor: f Osoba()
    ► [[Prototype]]: Object
>
```

2. Praksa je definirati svojstva unutar konstruktor funkcije objekta a metode na prototipu objekta

```

> function Osoba(starost) {
    this.godine = starost;
  }
< undefined

> Osoba.prototype.govor = function() {
    return 'Govorim...';
  };
< f () {
    return 'Govorim...';
  }

> const marica = new Osoba(25);
< undefined

> marica.godine
< 25

> marica.govor()
< 'Govorim...'

>

```

3. Nasljeđivanje primjer

```

> const Osoba = {
    govor() {
        return 'Govorim...';
    }
};
< undefined

> const ivica = Object.create(Osoba);
< undefined

> ivica.govor();
< 'Govorim...'

> const marica = {};
< undefined

> Object.setPrototypeOf(marica, Osoba);
< ► {}

> marica.govor();
< 'Govorim...'

>

```