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

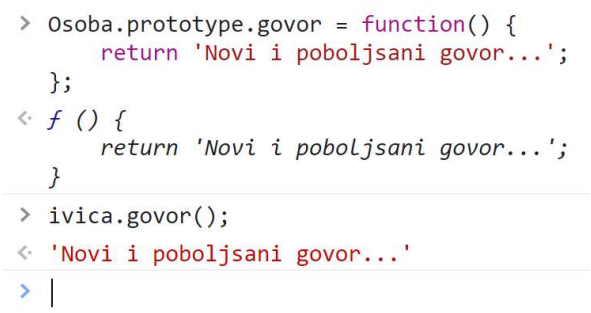
```
> ivica
<  Osoba {}
```
-->> ivica je objekt tipa Osoba, nema funkcije govor

Ako prikažemo cijelo stablo ispod objekta Osoba

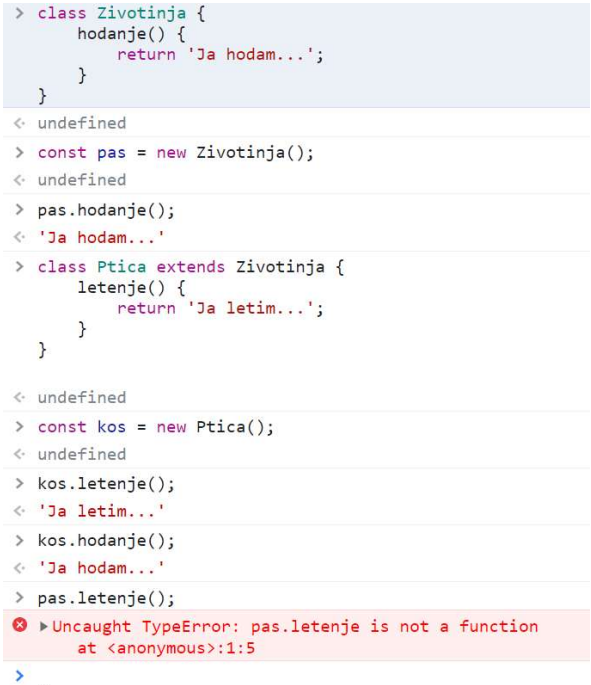
funkcija govor se nalazi na prototype objektu na osnovu kojeg je kreiran objekt ivica

5. Usporedba objekta ivica i klase Osoba

6. Ažuriranje funkcije govor klase Osoba



7. Primjer nasljeđivanja



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...';
  };
< ƒ () {
      return 'Govorim...';
  }
> const ivica = new Osoba();
< undefined
> ivica.govor();
< 'Govorim...'
> ivica
< ▼ Osoba {} i
    ▼ [[Prototype]]: Object
      ▶ govor: ƒ ()
      ▶ constructor: ƒ 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: ƒ} i
    ▶ govor: ƒ ()
    ▼ [[Prototype]]: Object
      ▶ constructor: ƒ 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...';
  };
< ƒ () {
      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...'
>
```