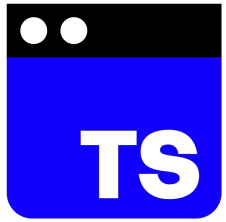
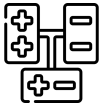


OOP
Paveldimumas

TypeScript

9 paskaita

Paskaitos eiga



Kas yra paveldimumas?

Paveldimumo tikslas

Paveldimumo įgyvendinimas

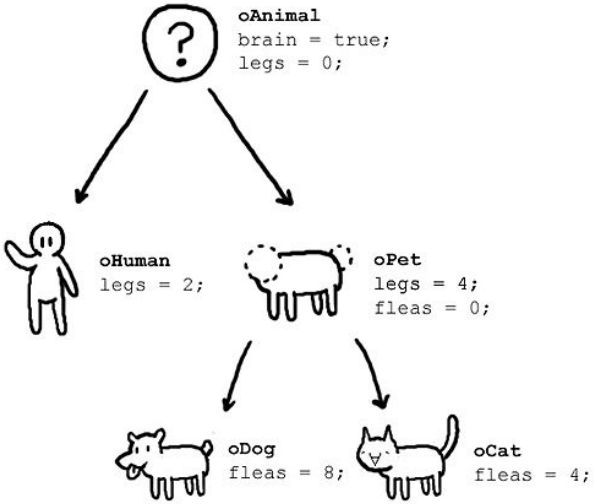
Kas yra paveldimumas?

Kas yra paveldimumas?

Paveldimumas tai ryšys tarp klasių, kuriame vaikinė klasė gali naudoti tėvinės klasės, savybes ir metodus.

Paveldimumo ryšys turėtų būti sudaromas tik tuo atveju kuomet galime pasakyti: “klasė A YRA klasė B”, pvz.:

- Beržas YRA medis
- Katė YRA gyvūnas
- Kompiuteris YRA Elektroninis įrenginys

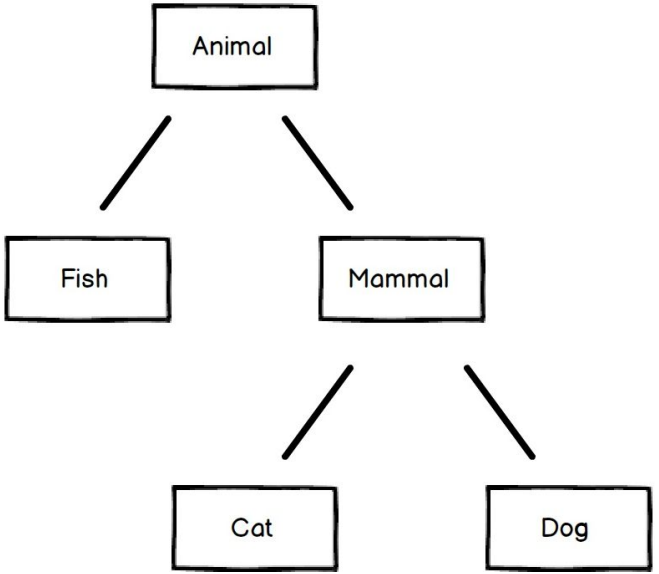


Kas yra paveldimumas?

Bet kokia klasė gali turėti tik vieną tėvinę klasę.

Galimas grandininis paveldimumas: klasė A paveldi klasę B, o klasė B paveldi klasę C, pvz.:

Katė yra Žinduolis, o Žinduolis yra Gyvūnas



Paveldimumo tikslas

Paveldimumo tikslas

Klasės formuojamos naudojant paveldimumą tam, kad būtų galima perpanaudoti bendras savybes ir metodus. Tai vienas iš “Don’t Repeat Yourself - **DRY**” programavimo paradigmos įgyvendinimo būdų.

Dažnai panašios struktūros dokumentus laikome vienoje kolekcijoje, pvz.: Dokumentai, Darbuotojai, Daržovės ir t.t.

Paveldimumo tikslas

Kuomet visi objektai paveldi tą pačią tėvinę klasė, galima jiems visiems įvykdyti metodus aprašytus tėvinėje klasėje, pvz.:

- Atspausdinti visus dokumentus
- Sumokėti visų darbuotojų atlyginimus
- Supjaustyti visas daržoves ir t.t.

Paveldimumo įgyvendinimas

Bendro funkcionalumo atpažinimas

Tam, kad kurti tėvinę, turime susidurti su situacija, kuomet skirtingų klasių objektai vykdo tokį patį kodą, pvz.:

```
class Lecturer {
  constructor(
    private name: string,
    private surname: string,
    private salary: number,
  ) {
    this.salary = salary;
  }

  get fullname() {
    return this.name + ' ' + this.surname;
  }
}
```

```
class Student {
  private marks: number[];

  constructor(
    private name: string,
    private surname: string,
  ) {
    this.marks = [];
  }

  get fullname() {
    return this.name + ' ' + this.surname;
  }

  get avg() {
    return this.marks.reduce((s, x) => s + x, 0) /
this.marks.length;
  }

  addMark(mark: number) {
    this.marks.push(mark);
  }
```

Bendro funkcionalumo aprašymas atskiroje klasėje

Tuomet atpažintą bendrą funkcionalumą galime aprašyti naujoje-atskiroje klasėje.
Naujos-atskiros klasės pavadinimas turėtų sufleruoti veiksmus, kurie joje yra aprašomi.

protected pasiekiamumo operatorius reiškia, kad savybė ar metodas bus pasiekiamas šios klasės viduje arba vaikinių klasių viduje.

```
interface PersonProps {
  name: string;
  surname: string;
}

class Person {
  protected name: string;
  protected surname: string;

  constructor({ name, surname }: PersonProps) {
    this.name = name;
    this.surname = surname;
  }

  get fullname() {
    return this.name + ' ' + this.surname;
  }
}
```

Paveldimumo ryšio įgalinimas

Sukūrus naują klasę ir iškėlus į ją bendrą funkcionalumą, galime vaikinėse klasėse pašalinti tarpusavyje pasikartojančias savybes ir metodus. Įgalinus paveldėjimą, galėsime pasiekti jas iš tėvinės klasės ir funkcionalumas nepasikeis.

Paveldimumo ryšio įgalinimas

Paveldimumo ryšiui sudaryti naudojamas raktažodis extends klasės deklaracijoje.

Tam, kad iškviesti tėvinės klasės konstruktorių naudojamas raktažodis super.

Tokiu būdu perpanaudojame tą patį kodą, vietoj to, kad kiekvienoje klasėje aprašytume atskirai.

```
class Student extends Person {
  private marks: number[];

  constructor(personProps: PersonProps) {
    super(personProps);
    this.marks = [];
  }

  get avg() {
    return this.marks.reduce((s, x) => s + x, 0) / this.marks.length;
  }

  addMark(mark: number) {
    this.marks.push(mark);
  }
}
```

```
class Lecturer extends Person{
  constructor(private salary: number, personProps: PersonProps) {
    super(personProps);
    this.salary = salary;
  }
}
```

Klausimai?

Paskaitos darbas

Paskaitos darbas

Paskaitoje atliksime užduotis, tokia eiga:

1. Sprendžiame užduotis savarankiškai
2. Po savarankiško sprendimo laiko (10-30 min.) dėstytojas išsprendžia 1 užduotį argumentuodamas sprendimą
3. Studentai užduoda klausimus apie sprendimą
4. Sprendimų palyginimas
5. Atliekama sekanti užduotis

Jeigu išsprendėte užduotį anksčiau nei kiti, spręskite sekančias užduotis.

Užduoties aptarimo metu, nesidrovėkite klausti kuo daugiau klausimų. Nebūtinai jūsų sprendimas yra prastesnis. Galbūt net geresnis?

Iki kito karto!