Objektinis programavimas: Intro

Paruošė Giedrius Graževičius

Šiek tiek istorijos

- Simula
  - 1967 Osle
  - Atsiranda klasė: *class*
  - Skirta simuliacijoms
- Smalltalk
  - 1972
  - Atsiranda *object-oriented programming*

Objektais paremtas

- Gali egzistuoti klasė
- Egzistuoja objektai - duomenys ir veiksmai viename
- Bet...

Objektiškai orientuotas

- Enkapsuliacija
- Paveldėjimas
- Polimorfizmas

Klasė

- Ką objektas žinos
- Ką objektas mokės

Ką objektas žino?

EIS GROUP™

```
public class Šuo {
    private String vardas;
    private Location maistas;
}
```

Ką objektas moka?

EIS GROUP™

```
public class Šuo {
    public void lok() {
        System.out.println(“Au au”);
         }
}
```

Kaip objektai gimsta?

EIS GROUP™

- Operatorius *new*

```
Šuo šuo1 = new Šuo();
Šuo šuo2 = new Šuo();

šuo2.lok();
```

Konstruktorius

EIS GROUP™

```
public class Šuo  {
}
// tapatu
public class Šuo  {
    public Šuo() {
    }
}
```

Konstruktorius (2)

EIS GROUP™

- Metodas
- Vardas sutampa su klasės
- Neturi grąžinamo tipo
- Gali būti keli (0..*)

```
public class Šuo  {
    public Šuo() {
    }
}
```

Konstruktorius (3)

EIS GROUP™

```
public class Šuo  {
    private String vardas;
    public Šuo(String vardas) {
        this.vardas = vardas;
    }
}

Šuo šuo = new Šuo(“Reksas”);
```

Konstruktorius (4)

EIS GROUP™

```
public class Šuo  {
    private String vardas;
    public Šuo() {}
    public Šuo(String vardas) {
        this.vardas = vardas;
    }
}
Šuo šuo = new Šuo(“Reksas”);
Šuo šuo2 = new Šuo();
```

Metodai (perkrova, *overloading*)

EIS GROUP™

```
public class Šuo  {
    public void lok() {
        System.out.println(“Au”);
    }
    public void lok(int kiekKartu) {
        for (int i = 0; i < kiekKartu; i++) {
            lok();
        }
    }
}
```

```
Šuo šuo = new Šuo();
šuo.lok();
šuo.lok(5);
```

Paveldėjimas (inheritance)

EIS GROUP™

- Vieni objektai iš kitų gali paveldėti
  - tai ką žino (laukus)
  - tai ką moka (metodus)
    - bet ne konstruktorius*

Paveldėjimas (2)

EIS GROUP™

```
public class Šuo {
    public void lok() { System.out.println("Au"); }
}
public class Spanielis extends Šuo {
}

Spanielis spanielis = new Spanielis();
spanielis.lok();
```

Paveldėjimas (3)

EIS GROUP™

```
public class Šuo {
    public String vardas;
}
public class Spanielis extends Šuo {
}

Spanielis spanielis = new Spanielis();
spanielis.vardas = “Reksas”;
```

Enkapsuliacija

- Veiksmai ir duomenys supakuoti kartu
- Programuotojas renkasi ką rodyti, o ką slėpti nuo objekto naudotojo (kito programuotojo)

Enkapsuliacija (2)

EIS GROUP™

- *public* - visi mato
- *private* - mato tik pats objektas
- *protected* - mato tik pats objektas, objektai kurie paveldi ir paketas
- *package* - mato paketas (naudojama pagal nutylėjimą)

Enkapsuliacija (3)

EIS GROUP™

```
public class Šuo {
    private String vardas;
}

Šuo šuo = new Šuo();
šuo.vardas = “Reksas”; // Nesikompiliuos, nes laukas nėra
  matomas už klasės ribų
```

Enkapsuliacija (4)

EIS GROUP™

```
public class Šuo {
    private String vardas;
}

Šuo šuo = new Šuo();
šuo.vardas = “Reksas”; // All is fine, nes laukas public
```

Enkapsuliacija (5)

EIS GROUP™

```
public class Šuo {
    private String vardas;
}
public class Spanielis extends Šuo {
    public String grąžinkVardą() {
        return vardas; // Nesikompiliuos, nes...
    }
}
```

Enkapsuliacija (6)


```
public class Šuo {
    protected String vardas;
}
public class Spanielis extends Šuo {
    public String grąžinkVardą() {
        return vardas; // All is fine
    }
}
```

Praktika

EIS GROUP™

- Apibrėžti klasę *Sąskaita* su savybėmis
  - Numeris
  - Savininkas
  - Balansas
- ir metodais
  - leidžiančiais įnešti pinigų sumą
  - leidžiančiais pasiimti pinigų sumą

Praktika (2)

EIS GROUP™

- Reikalavimai
  - Sąskaitos numeris ir savininkas turi būti nekeičiami, bet turi būti įmanoma juos sužinoti
  - Iš sąskaitos turi būti negalima išimti daugiau pinigų nei joje yra įdėta (turi būti būdas sužinoti ar išėmimo operacija pavyko ar ne)