OOP - Úvod, Třídy, Objekt

PVA2 Programování a vývoj aplikací

Adam Fišer

Obsah

Objektově orientované programování

- Doposud jsme programovali v procedurálním stylu
- Objektově orientované programování (OOP) je programovací paradigma
- Pomocí OOP se snažíme modelovat reálný svět
- Hlavními koncepty OOP jsou dědičnost, zapouzdření a polymorfizmus

OOP

- V OOP se snažíme rozdělit program do samostatných částí **tříd** a tyto části spolu komunikují pomocí **metod** a **vlastností**
- OOP je založeno na objektech tj. strukturách, které obsahují jak informace (data), tak funkce (metody), které s těmi daty umí pracovat.
- Hlavním konceptem OOP je svázat data a funkce, které s nimi pracují, do jednoho celku tak, aby k těmto datům neměla přístup žádná jiná část kódu aplikace.
- V Pythonu je všechno objekt. Přesněji řečeno: všechno, co můžeme uložit do Pythoní proměnné – každá hodnota – je objekt. Čísla, řetězce, funkce, seznamy, soubory, metody, moduly, třídy, matice – všechno jsou objekty. (Nemusí a obecně neplatí pro jiné jazyky)

Třída

Třída

- Třída je základním stavebním prvkem OOP.
- Jedná se o šablonu, která definuje vlastnosti a chování určitého typu objektu.
- Každá třída se zapisuje do samostatného souboru, který je pojmenován stejně, jako název třídy.

> Třída je šablona, podle které vytváříme objekty

- Před název třídy se používá klíčové slovo `class`
- Název tříd se píše s velkým počátečním písmenem
- Jedna třída = jeden soubor, nelze ji rozdělit mezi více.
- Třída musí být unikátní. V celé aplikaci může existovat pouze jednou.
- Instancí může existovat neomezené množství. Jsme limitováni pouze HW zdroji.

Třída

- Atributy: Proměnné (property) definující vlastnosti objektu.
- Metody: Funkce nebo procedury, které provádějí určité akce nebo manipulují s atributy třídy.
- Konstruktor: Speciální metoda, která se volá při vytváření nového objektu a inicializuje jeho atributy.
- Destruktor: Speciální metoda, která se volá při zániku objektu a může provádět úklidové operace.

Ukázka třídy `Auto.py`

```
# Vytvoření třídy Auto
# Klíčové slovo class <názevTřídy>
# Název třídy - první písmeno velké
# Název souboru odpovídá názvu třídy, standardně v jednoném čísle

class Auto:
    # Obsah třídy
```

Atributy

- Atributy jsou proměnné, které náleží objektu.
- Atributy se definují v konstruktoru třídy.

Ukázka atributů

```
class Auto:
    # Atribut třídy
    atribut = "hodnota"

    znacka = "Škoda"

    # Prázdná hodnota
    model = None
```

Modifikátory přístupnosti

- Atributy mohou být:
  - veřejné `public`,
  - chráněné `protected` prefix `_`
  - nebo privátní `private` prefix `__`
- Veřejné atributy jsou dostupné zvenčí, chráněné pouze z třídy a dědičných tříd, privátní pouze z třídy.

```
1   class Auto:
2       # public
3       znacka = "Škoda"
4
5       # protected
6       _prodejniCena = 790000
7
8       # private
9       __vyrobniCena = 440000
```

Konstanty

- jsou atributy, jejichž hodnota se nemění
- definují se jako privátní atributy a pojmenovávají se velkými písmeny
- před pojmenováním je uvedeno klíčové slovo `const`
- definují se v těle třídy, mimo metody
- jsou dostupné pouze z třídy, ne z objektu
- volají se pomocí názvu třídy

Ukázka kontant

```
1   class Auto:
2       const MAX_RYCHLOST = 250
```

Třída - příklad

```python
class Auto:
    # Atributy
    znacka = None
    model = None
    rok_vyroby = None

    # Konstruktor
    def __init__(self, znacka, model, rok_vyroby):
        self.znacka = znacka
        self.model = model
        self.rok_vyroby = rok_vyroby

    # Metoda
    def info(self):
        return f"{self.znacka} {self.model} z roku {self.rok_vyroby}"
```


Objekt

Objekt

- **Objekt je konkrétní instance třídy**
- má vlastnosti (atributy) a chování (metody), které jsme nadefeinovali v třídě
- Objekty mohou spolupracovat mezi sebou
- Objekty mohou být:
  - vnořené (objekt v objektu)
  - děděné (třída může dědit od jiné třídy)
  - polymorfní (objekty mohou mít stejné rozhraní, ale jinou implementaci)
  - zapouzdřené (skrýváme vnitřní stav objektu)
- Python je objektově orientovaný programovací jazyk, takže se v něm se vším zachází jako s objektem. Objekt je entita reálného života. Je to soubor různých dat a funkcí, které s těmito daty pracují.

Objekt - příklad

```
# Vytvoření objektu

# Vytvoření objektu auto1, instance třídy Auto
auto1 = Auto()

# Vytvoření objektu auto2, instance třídy Auto s využitím konstruktoru
auto2 = Auto("Škoda", "Octavia", 2015)
```

Třída vs Objekt

- Objekt je něco "živého", co právě teď existuje tzn. má instanci, může něco vykonávat nebo reagovat na jiné objekty.
- Třída je staticky napsaný kód, který se bude teprve vyhodnocovat a po celou dobu zůstává stejný. Jaká si šablona popisující objekt.
- Třída je uživatelem definovaná datová struktura, která spojuje datové členy a metody do jednoho celku.
- Třída je plán/šablona kódu pro vytváření objektů.
- Pomocí třídy můžete vytvořit libovolný počet objektů. Jsme limitování pouze HW zdroji.

Více současných instancí stejné třídy

- Každý objekt má svůj lokální kontext, který platí v jeho vnitřnostech.
- Můžeme vytvořit mnoho nezávislých instancí té samé třídy a s těmi libovolně manipulovat.
- Instancí můžeme vytvořit dokonce dynamický počet a například je uložit jako prvky pole.
- Všechny instance jsou uloženy v operační paměti – pozor na disponibilní zdroje.

```
1   #objekt = Třída()
2   pes = Pes()
3
4   druhyPes = Pes()
5
6   kolie = Pes()
```

Více současných instancí stejné třídy

```
pavla = Zamestnanec()
pavla.jmeno = "Pavla"

zuzka = Zamestnanec()
zuzka.jmeno = "Zuzana"

zam = Zamestnanec()
zam.jmeno = "Igor"
```

Nebo lépe, s využitím kolekce:

```
zamestnanci = []
zamestnanci.append( Zamestnanec("Pavla") )
zamestnanci.append( Zamestnanec("Zuzana") )
zamestnanci.append( Zamestnanec("Igor") )
```

Děkuji za pozornost

Otázky?

Repository / Prezentace