

Програмування-1

Лекція 6

Основи ООП

Декларування класів

- Декларування класів

```
[<модифікатори>] class <НазваКласу>  
    [extends <НазваБазовогоКласу>]  
    [implements <НазваІнтерфейсу1>[, <НазваІнтерфейсу2> [, ...]]]  
{  
    <декларація полів, конструкторів та методів>  
}
```

- Декларування полів

```
[<модифікатори>] <тип> <назваПоля> [=<значення_за_замовчуванням>];
```

- Декларація методів

```
[<модифікатори>] <тип> <назваМетода>([<список параметрів>])  
    [throws <список класів виняткових ситуацій>]{  
    <програмний код>  
}
```

Статичні поля та методи

- Статичні поля та методи належать всьому класу, а не об'єктам
 - з ними можна працювати без створення об'єктів
 - початківці та ледарі за це їх дуже люблять. НЕ РОБІТЬ ТАК!
 - використовуйте **static** лише там, де це дійсно необхідно
- Статичні методи не прив'язані до конкретних об'єктів, тому:
 - з них можна звертатись лише до статичних полів та методів класу
 - з них неможна звертатись до нестатичних полів та методів класу
 - в них відсутній **this** (див. нижче)
- Ззовні до **static** полів та методів можна звертатись так:
 - через ім'я класу (**НазваКласу.статичнеПоле**)
 - робіть саме так
 - через посилання на об'єкта (**посилання.статичнеПоле**)
 - ніколи так не робіть
- Для **static** методів не працює поліморфізм
 - статичні методи «прив'язані» до класів, та не «прив'язані» до об'єктів
 - вибір статичного метода відбувається на етапі компіляції, а не під час виконання

Статичні поля та методи - приклад

```
12 public class Demo {
13     int a;
14     static int b;
15
16     public static void main(String[] args) {
17         Demo demo1 = new Demo();
18         Demo demo2 = new Demo();
19
20         demo1.a++;
21         demo1.b++; // Ugly style!!! Use: Demo.b++
22         System.out.println("demo1: " + demo1.a + " " + demo1.b);
23
24         demo2.a++;
25         demo2.b++; // Ugly style!!! Use: Demo.b++
26         System.out.println("demo2: " + demo2.a + " " + demo2.b);
27     }
28 }
```

Вивод - JavaApplication2 (run) ✖

```
run:
demo1: 1 1
demo2: 1 2
```

Пам'ять

Клас Demo

static int b;

Об'єкт demo1

int a;

Об'єкт demo2

int a;

Перевантаження методів (overload)

- **Перевантаження методів (overload)** – декларація декількох методів з однаковими назвами, але різними наборами параметрів
 - не плутати з **override** (перевизначення у підкласі метода суперкласу)
- **Сигнатура методу** – **назва методу + тип параметрів**
 - Тип значення що повертається не є частиною сигнатури
- **Вибір** потрібного **методу** виконується **на етапі компіляції** по **сигнатурі методу**
- При перевантаженні **не може бути** двох методів з однаковою сигнатурою