

Programowanie obiektowe

Klasy

Obiekty

Metody i konstruktory

Klasy (demo)

Jak zapisać w programie informacje opisujące Osobę (Jana Kowalskiego,
w wieku 25 lat z Wrocławia)?

Klasy

- Klasa jest **szablonem** pozwalającym definiować własne, bardziej złożone typy danych, np.:
 - Osoba (imię, nazwisko, wiek)
 - Samochód (marka, model, moc silnika)
 - Produkt (nazwa, producent, cena)
- Klasa pozwala zdefiniować cechy oraz funkcjonalności



- imię?
- nazwisko?
- wiek?
- miejsce zamieszkania?

Klasy – przykład (demo)

- Klasa Person posiada cztery **pola klasy** (firstName, lastName, age, city)
- Pola klasy podobnie jak zmienne muszą mieć określony **typ** oraz **nazwę**.
- Klasa staje się automatycznie nowym typem danych, którego możemy używać w naszej aplikacji

```
class Person {  
    String firstName;  
    String lastName;  
    int age;  
    String city;  
}
```

Obiekty

Klasa



Obiekt – konkretny egzemplarz klasy



Obiekty

Obiekty tworzymy podobnie do zwykłych zmiennych – poprzez deklarację i inicjalizację. Klasy traktujemy jako nowe, bardziej złożone typy danych.

Inicjalizacja polega na utworzeniu obiektu przez wywołanie operatora **new**

Do składowych obiektu odwołujemy się przez operator kropki .

Na podstawie klasy można tworzyć dowolną ilość obiektów

```
class Person {  
    String firstName;  
    String lastName;  
    int age;  
    String city;  
}
```

```
public static void main(String[] args) {  
    Person person1 = new Person();  
    System.out.println(person.firstName);  
}
```

Obiekty (demo)

Obiekty po utworzeniu mają do pól przypisane wartości domyślne:

- typy liczbowe – 0 lub 0.0
- boolean – false
- char – '\u0000'
- String i inne typy obiektowe – null

W celu przypisania nowych wartości również robimy to odwołując się przez operator kropki.

Pola obiektu zachowują się jak zmienne – można je modyfikować o ile nie są poprzedzone słowem **final**.

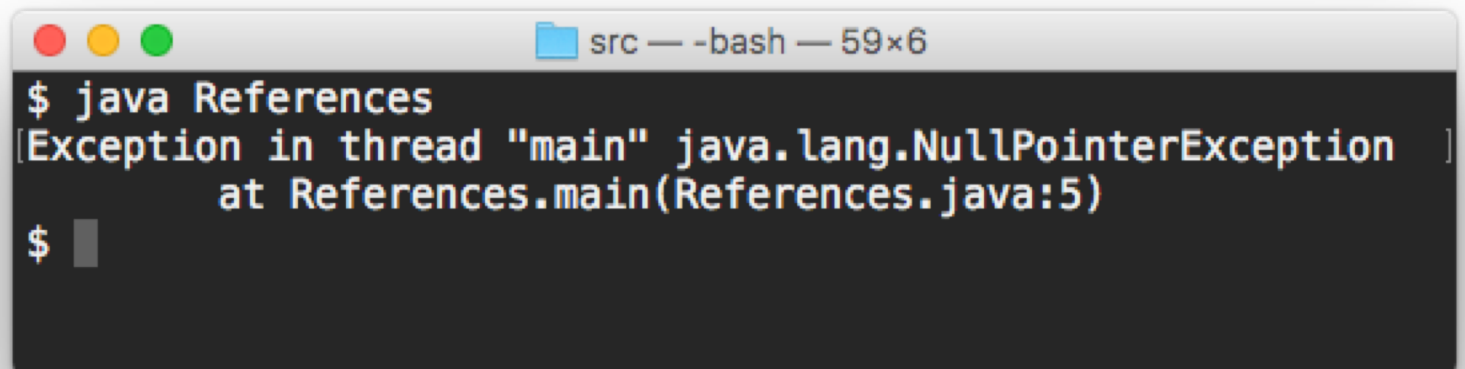
```
public static void main(String[] args) {  
    Person person1 = new Person();  
    person1.firstName = "Jan";  
    person1.lastName = "Kowalski";  
    person1.age = 25;  
    person1.city = "Wrocław";  
  
    System.out.println(person.firstName);  
    System.out.println(person.lastName);  
    System.out.println(person.age);  
    System.out.println(person.city);  
  
    person1.firstName = "Andrzej";  
    System.out.println(person1.firstName);  
}
```

NullPointerException

Do zmiennych typów obiektowych możemy przypisać ich domyślną wartość – null, czyli pustą referencję

Jeśli spróbujesz odwołać się do dowolnej wartości poprzez operator kropki otrzymasz błąd NullPointerException

```
class References {  
    public static void main(String[] args) {  
        Person person1 = null;  
  
        System.out.println(person1.firstName);  
    }  
}
```

A terminal window with a title bar showing 'src' and '-bash' with a window size of '59x6'. The terminal has a dark background with light-colored text. It shows the command '\$ java References' followed by an exception message: '[Exception in thread "main" java.lang.NullPointerException]' and the location 'at References.main(References.java:5)'. A new prompt '\$' is visible on the next line.

```
src — -bash — 59x6  
$ java References  
[Exception in thread "main" java.lang.NullPointerException ]  
    at References.main(References.java:5)  
$
```


Pola typów obiektowych

Klasy mogą posiadać pola innych typów obiektowych. Pola takie domyślnie również przyjmują wartość null.

```
public class Address {  
    String street;  
    String city;  
    String postalCode;  
}
```

```
class Person {  
    String firstName;  
    String lastName;  
    int age;  
    Address address;  
}
```

```
public class People {  
  
    public static void main(String[] args) {  
        Person person1 = new Person();  
        person1.address = new Address();  
        person1.address.city = "Wrocław";  
        //...  
    }  
  
}
```

Ćwiczenie

Obiekty a referencje – przypadek 1

```
Person person1 = new Person();
```

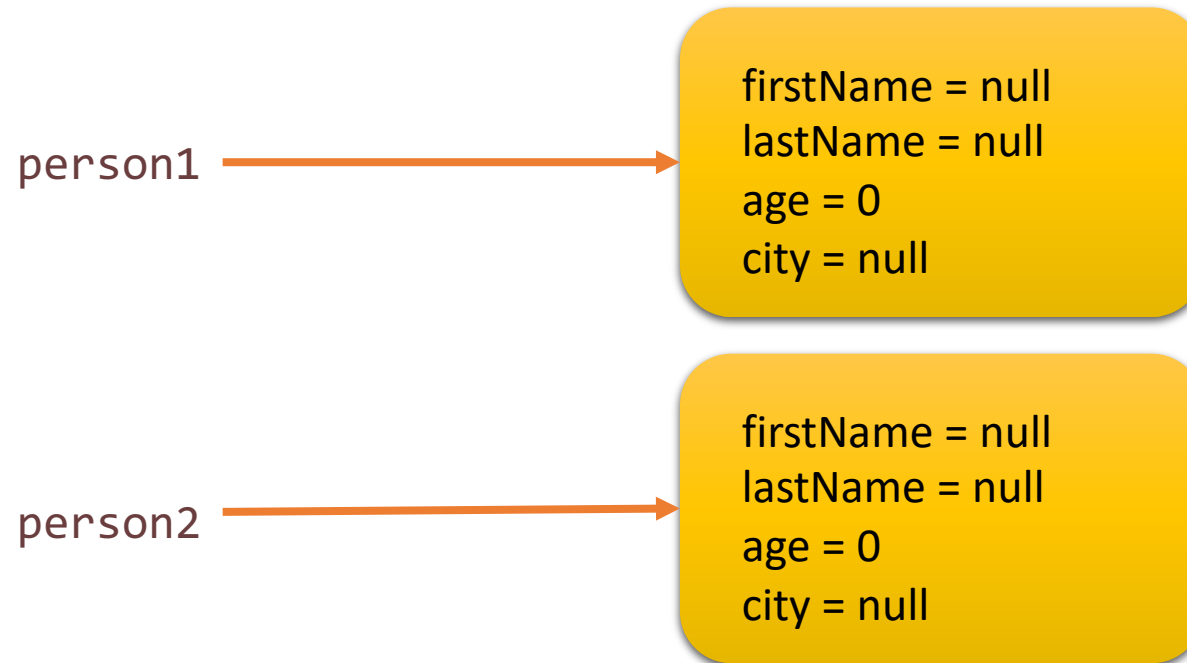
person1



firstName = null
lastName = null
age = 0
city = null

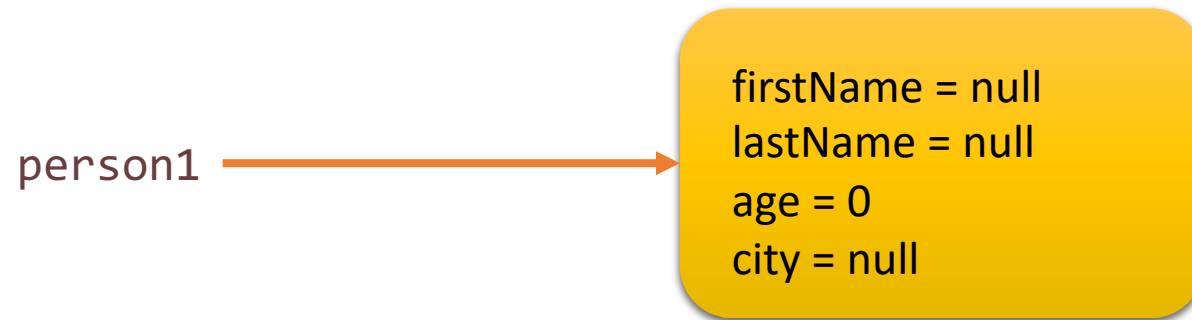
Obiekty a referencje – przypadek 1

```
Person person1 = new Person();  
Person person2 = new Person();
```



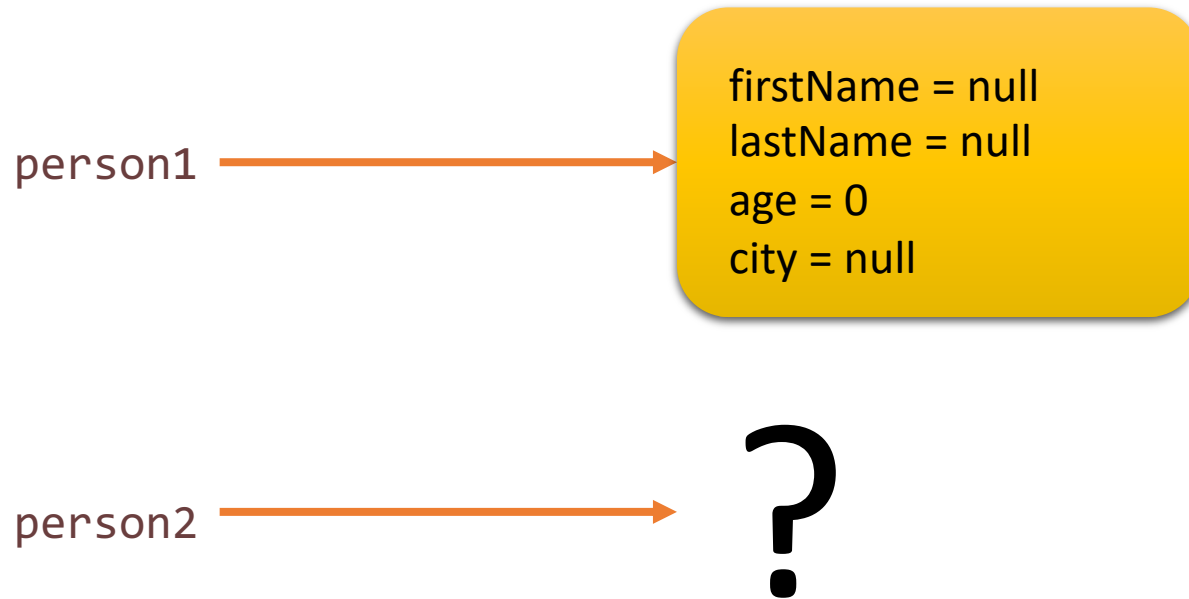
Obiekty a referencje – przypadek 2

```
Person person1 = new Person();
```



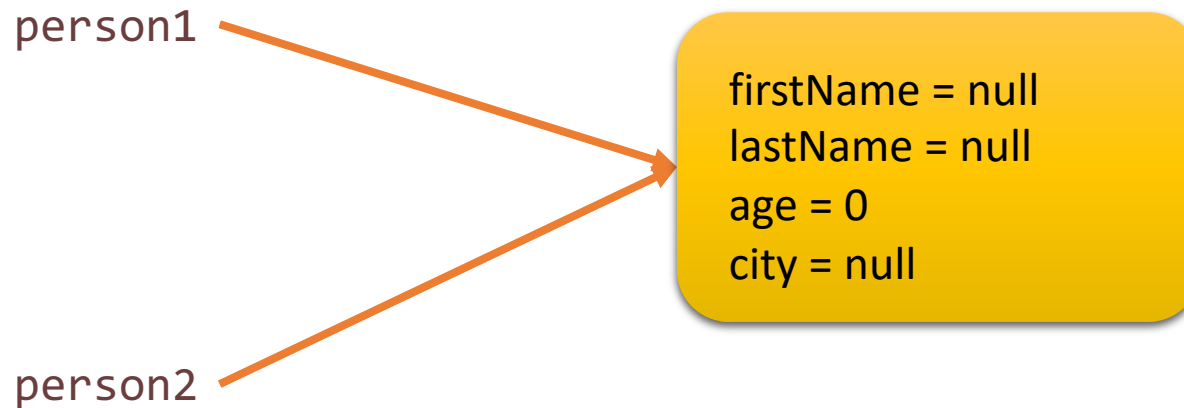
Obiekty a referencje – przypadek 2

```
Person person1 = new Person();  
Person person2 = person1;
```



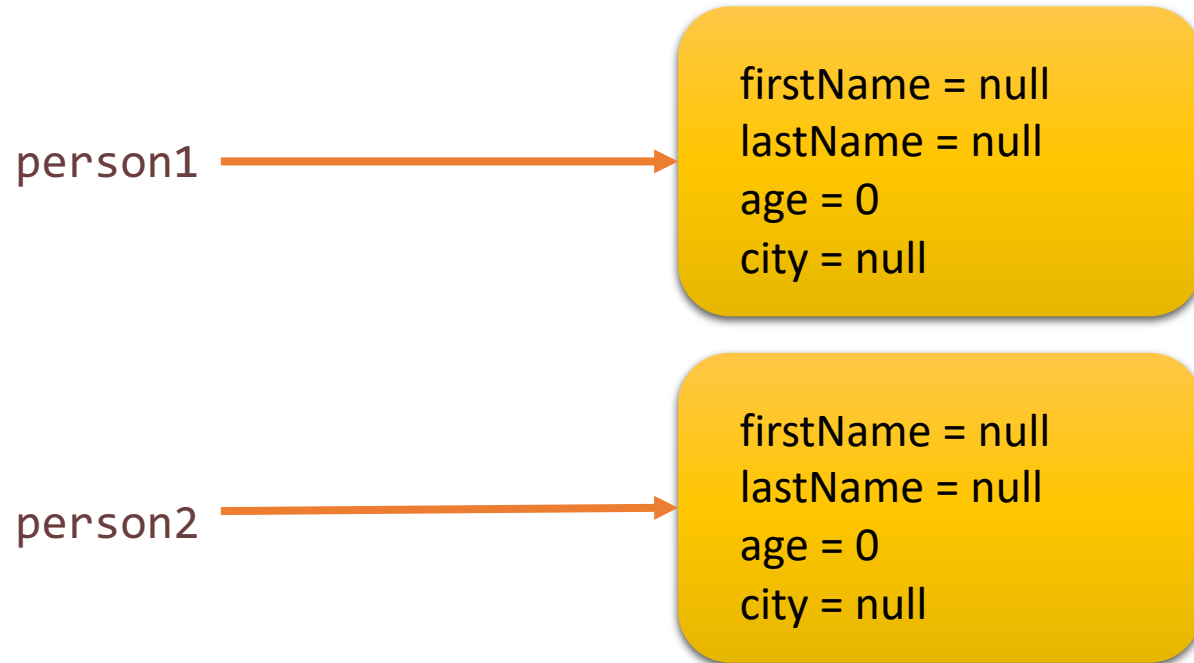
Obiekty a referencje – przypadek 2

```
Person person1 = new Person();  
Person person2 = person1;
```

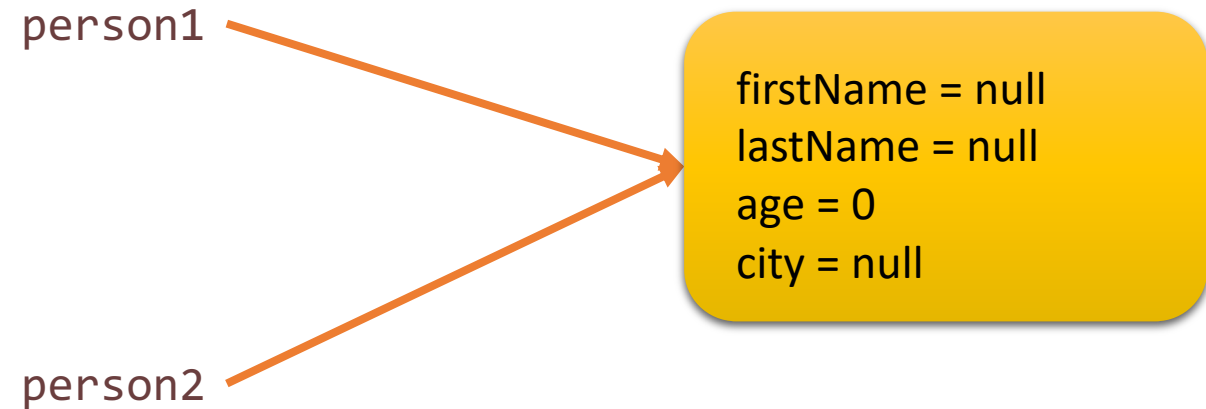


Obiekty a referencje – przypadek 1 i 2

```
Person person1 = new Person();  
Person person2 = new Person();
```



```
Person person1 = new Person();  
Person person2 = person1;
```



Obiekty a referencje (1) (demo)

```
class References {  
    public static void main(String[] args) {  
        Person person1 = new Person();  
        person1.firstName = "Jan";  
  
        Person person2 = new Person();  
        person2.firstName = "Andrzej";  
  
        System.out.println(person1.firstName);  
        System.out.println(person2.firstName);  
    }  
}
```

person1



firstName = Jan
lastName = null
age = 0
city = null

Obiekty a referencje (1) (demo)

```
class References {  
    public static void main(String[] args) {  
        Person person1 = new Person();  
        person1.firstName = "Jan";  
  
        Person person2 = new Person();  
        person2.firstName = "Andrzej";  
  
        System.out.println(person1.firstName);  
        System.out.println(person2.firstName);  
    }  
}
```

person1



firstName = Jan
lastName = null
age = 0
city = null

person2



firstName = Andrzej
lastName = null
age = 0
city = null

```
src — -bash — 36x6  
$ java References  
Jan  
Andrzej  
$
```

Obiekty a referencje (2) (demo)

```
class References {  
    public static void main(String[] args) {  
        Person person1 = new Person();  
        person1.firstName = "Jan";  
  
        Person person2 = person1;  
        person2.firstName = "Andrzej";  
  
        System.out.println(person1.firstName);  
        System.out.println(person2.firstName);  
    }  
}
```

person1



firstName = Jan
lastName = null
age = 0
city = null

Obiekty a referencje (2) (demo)

```
class References {  
    public static void main(String[] args) {  
        Person person1 = new Person();  
        person1.firstName = "Jan";  
  
        Person person2 = person1;  
        person2.firstName = "Andrzej";  
  
        System.out.println(person1.firstName);  
        System.out.println(person2.firstName);  
    }  
}
```

person1

person2

firstName = Jan
lastName = null
age = 0
city = null

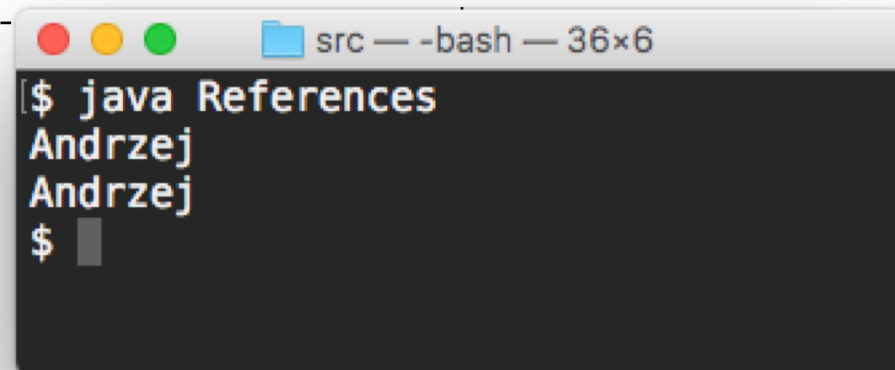
Obiekty a referencje (2) (demo)

```
class References {  
    public static void main(String[] args) {  
        Person person1 = new Person();  
        person1.firstName = "Jan";  
  
        Person person2 = person1;  
        person2.firstName = "Andrzej";  
  
        System.out.println(person1.firstName);  
        System.out.println(person2.firstName);  
    }  
}
```

person1

person2

firstName = Andrzej
lastName = null
age = 0
city = null



```
src — -bash — 36x6  
[$ java References  
Andrzej  
Andrzej  
$
```

Ćwiczenie

Ćwiczenie*

Konstruktory

- Każda klasa posiada konstruktor domyślny, który wywołujemy przez **new NazwaKlasy()**
- Konstruktor domyślny tworzy obiekt i inicjuje pola wartościami domyślnymi

```
class Person {  
    String firstName;  
    String lastName;  
    int age;  
    String city;  
}
```

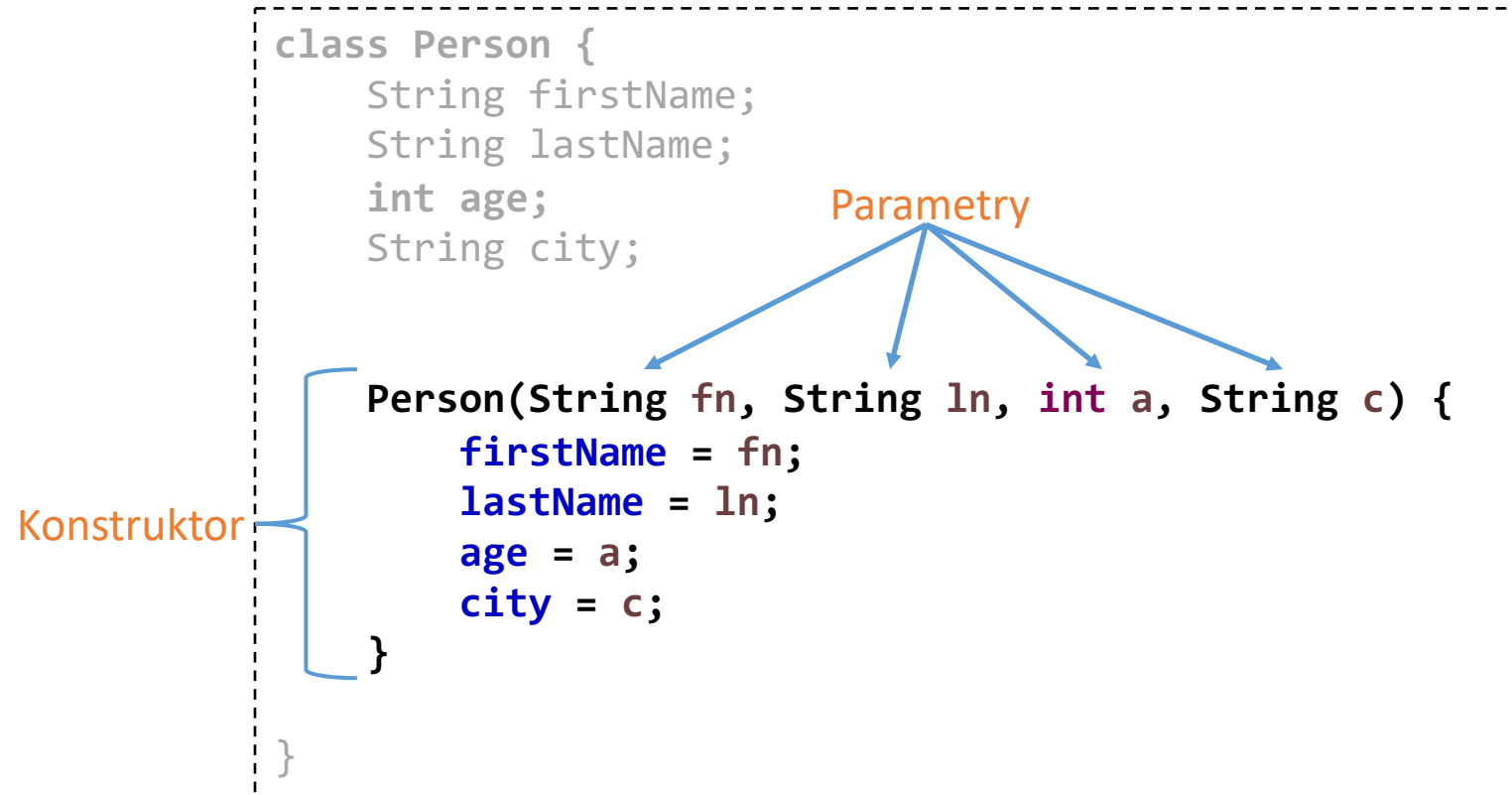
=

Konstruktor

```
class Person {  
    String firstName;  
    String lastName;  
    int age;  
    String city;  
  
    Person() {  
    }  
}
```


Konstruktory

- Możemy zdefiniować własny konstruktor, który uprości tworzenie obiektu
- **Jeśli zdefiniujemy jakikolwiek własny konstruktor, to konstruktor domyślny nie będzie wygenerowany**
- Konstruktor może mieć dowolną liczbę parametrów. Parametry są podobne do zmiennych, określamy je przez typ i nazwę i wymieniamy po przecinku w nawiasie okrągłym, a następnie przekazujemy przy wywoływaniu konstruktora przez **new**



Konstruktory (demo)


```
class Person {  
    String firstName;  
    String lastName;  
    int age;  
    String city;  
  
    Person(String fn, String ln, int a, String c) {  
        firstName = fn;  
        lastName = ln;  
        age = a;  
        city = c;  
    }  
}
```

```
class PersonTest2 {  
  
    public static void main(String[] args) {  
        Person person = new Person("Jan", "Kowalski", 25, "Wrocław");  
        System.out.println(person.firstName);  
        System.out.println(person.lastName);  
        System.out.println(person.age);  
        System.out.println(person.city);  
    }  
}
```

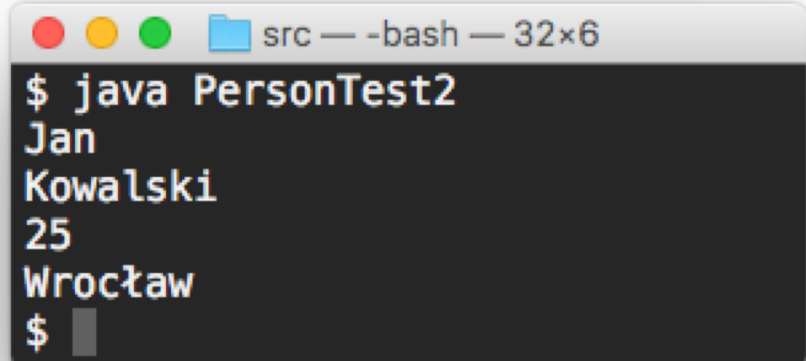
Konstruktory (demo)

```
class PersonTest2 {  
  
    public static void main(String[] args) {  
        Person person = new Person("Jan", "Kowalski", 25, "Wrocław");  
        System.out.println(person.firstName);  
        System.out.println(person.lastName);  
        System.out.println(person.age);  
        System.out.println(person.city);  
    }  
}
```

person



firstName = "Jan"
lastName = "Kowalski"
age = 25
city = "Wrocław"



```
src — -bash — 32x6  
$ java PersonTest2  
Jan  
Kowalski  
25  
Wrocław  
$
```

Ćwiczenie