
Wprowadzenie do



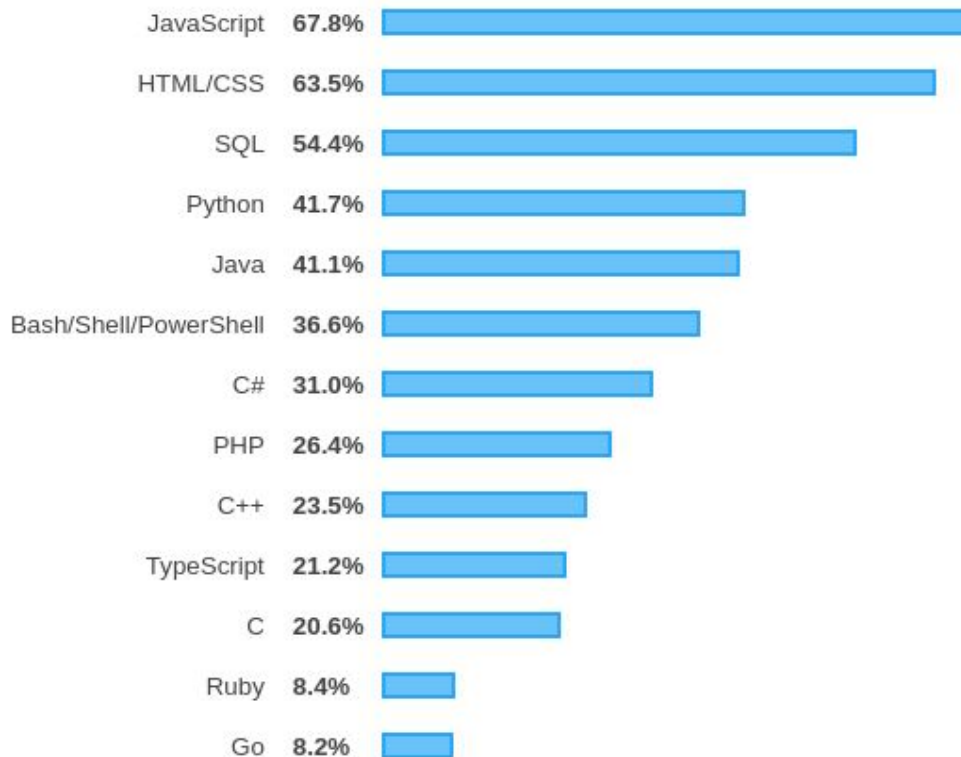
— dr inż. Wojciech Frącz —
fracz@agh.edu.pl

<https://fracz.com/mwo-java2>



Popularność Javy (marchewka)

Top 10



Quiz!

```
public class Elementary {  
    public static void main(String args[]) {  
        System.out.println(12345 + 54321);  
    }  
}
```



**WHEN I WROTE THIS, ONLY GOD AND I
UNDERSTOOD WHAT I WAS DOING**

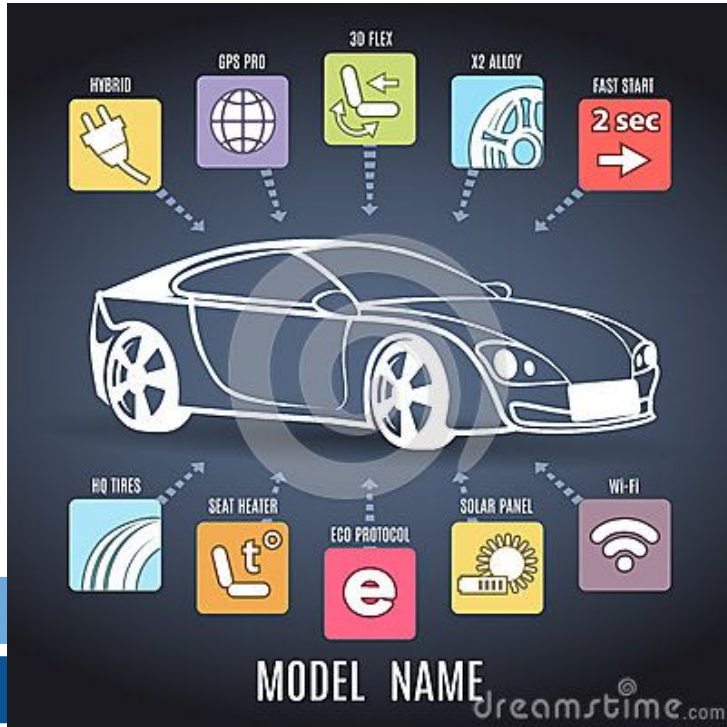
**NOW, GOD ONLY
KNOWS**

Programowanie obiektowe

- pozwala na odwzorowanie rzeczywistych bytów (rzeczy) w kodzie źródłowym
- pomaga na lepsze zrozumienie problemu
- pomaga lepiej zaimplementować rozwiązanie
- pozwala na dekompozycję problemu na mniejsze ("dziel i zwyciężaj")
- spaghetti code
 - https://pl.wikipedia.org/wiki/Spaghetti_code

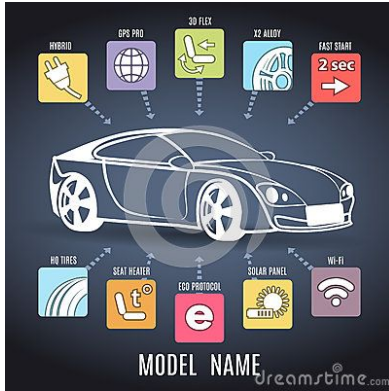


Programowanie obiektowe



```
class Car {  
  
}
```

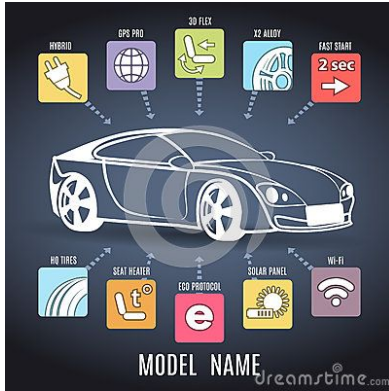
Programowanie obiektowe



```
class Car {
    void zapal() { System.out.println("Wrr"); }
}
```



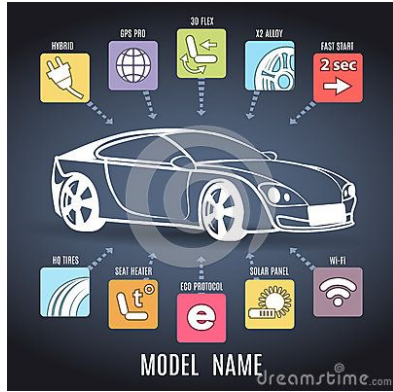
Programowanie obiektowe



```
class Car {  
    void zapal() { System.out.println("Wrr"); }  
    void zgas() { System.out.println("Psss"); }  
}
```



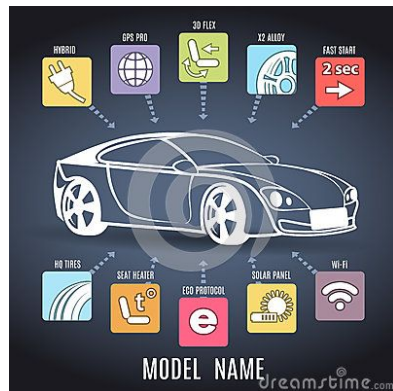
Programowanie obiektowe



```
class Car {
    void zapal() { System.out.println("Wrr"); }
    void zgas() { System.out.println("Psss"); }
    void jedz() { System.out.println("Brum"); }
}
```



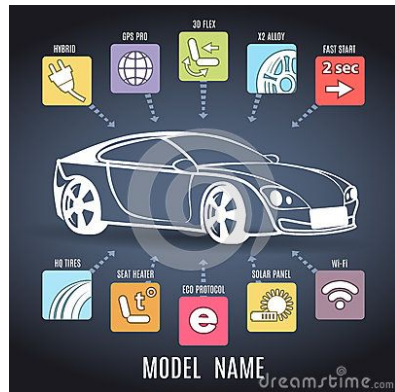
Programowanie obiektowe



```
class Car {
    void zapal() { System.out.println("Wrr"); }
    void zgas() { System.out.println("Psss"); }
    void jedz() { System.out.println("Brum"); }
    void stoj() { System.out.println("Piii"); }
}
```

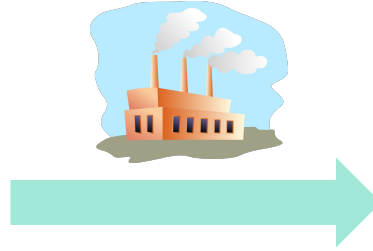


Programowanie obiektowe



```
class Car {
    void zapal() { System.out.println("Wrr"); }
    void zgas() { System.out.println("Psss"); }
    void jedz() { System.out.println("Brum"); }
    void stoj() { System.out.println("Piii"); }
    void skrecWLewo() {System.out.println("W lewo");}
    void skrecWPrawo() {System.out.println("W prawo");}
}
```

Programowanie obiektowe



```
class Car {  
}
```

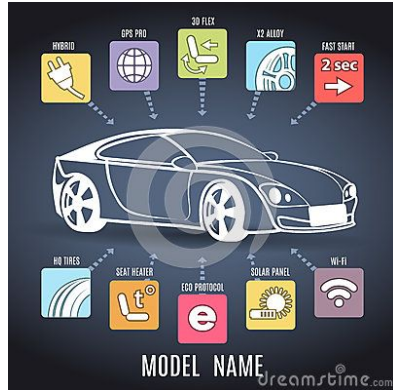
klasa



```
Car myCar = new Car();
```

obiekt (instancja klasy)

Programowanie obiektowe



```
class Car {  
}
```

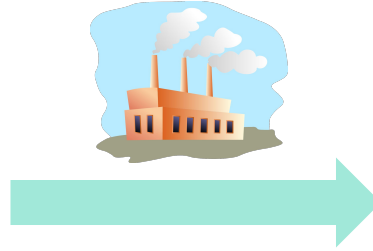
klasa



```
Car car1 = new Car();  
Car car2 = new Car();  
Car car3 = new Car();
```

obiekty (instancje klasy)

Programowanie obiektowe



```
class Car {  
}
```



```
Car car1 = new Car();
```

```
Car car2 = new Car();
```

```
Car car3 = new Car();
```

```
car2.jedz();
```

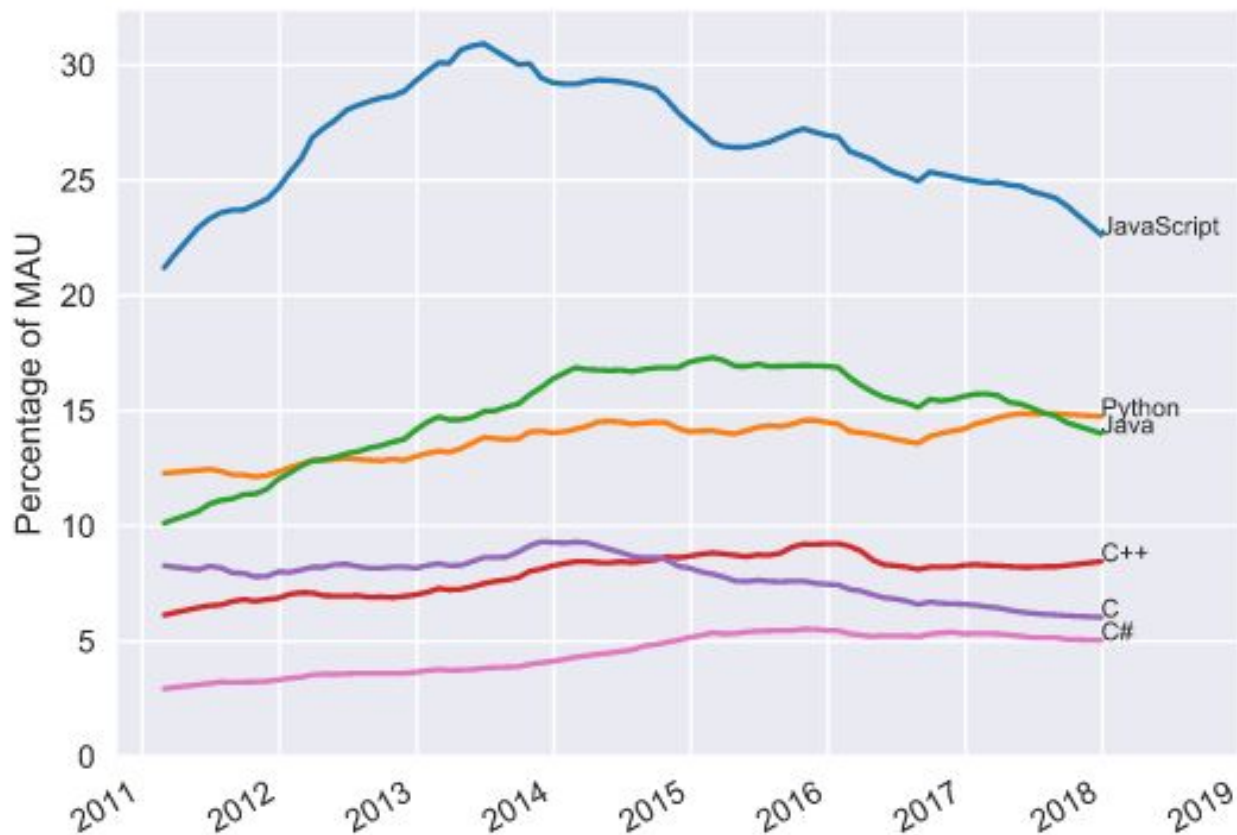
Quiz!

```
class Car {
    void zapal() { System.out.println("Wrr"); }
    void zgas() { System.out.println("Psss"); }
    void jedz() { System.out.println("Brum"); }
    void stoj() { System.out.println("Piii"); }
    void skrecWLewo() {System.out.println("W lewo");}
    void skrecWPrawo() {System.out.println("W prawo");}
}
```



```
public class MyGarage {
    public static void main(String[] args) {
        Car myCar = new Car();
        myCar.zapal();
        myCar.jedz();
        myCar.skrecWPrawo();
        myCar.stoj();
        myCar.zgas();
    }
}
```

Marchewka



Dekompozycja



- samochód nie jeździ sam
 - ma silnik
- nie kieruje sam
 - ma kierownicę
- różne podzespoły mogą być wykonywane przez różnych producentów
- te same podzespoły mogą być wykorzystywane w różnych samochodach

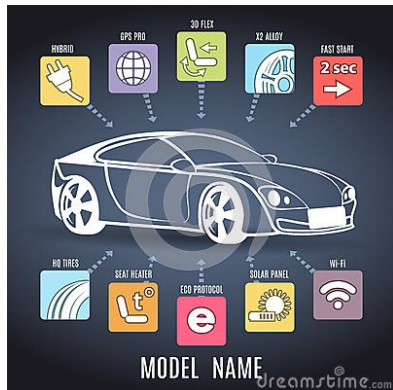
Dekompozycja



```
class Car {  
    Silnik silnik;  
    Kierownica kierownica;  
}
```



Dekompozycja



```
class Car {
    Silnik silnik;
    Kierownica kierownica;
    void zapal() { silnik.zapal(); }
    void zgas() { silnik.odetnijDoplywPaliwa(); }
    void jedz() { silnik.wiecejGazu(); }
    void stoj() { silnik.mniejGazu() }
    void skrecWLewo() { kierownica.lewo(); }
    void skrecWPrawo() { kierownica.prawo(); }
}
```

Dekompozycja

```
class Kierownica {
    void lewo() { System.out.println("W lewo"); }
    void prawo() { System.out.println("W prawo");}
}
```

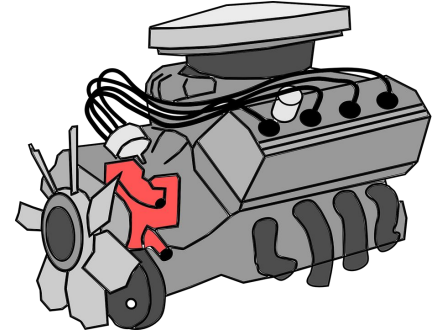


Dekompozycja

```
class Kierownica {
    void lewo() { System.out.println("W lewo"); }
    void prawo() { System.out.println("W prawo");}
    void reguluj() { /* ... */ };
}
```



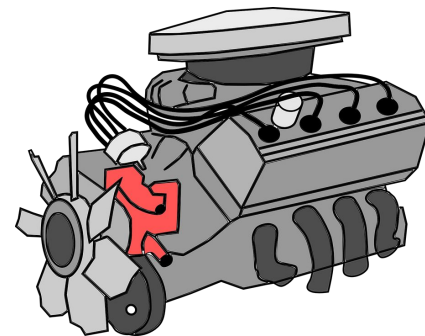
Dekompozycja



```
class Silnik {
    void zapal() { System.out.println("Wrrr"); }
    void odetnijDoplywPaliwa() { System.out.println("Psss"); }
    void wiecejGazu() { System.out.println("Brum"); }
    void mniejGazu() { System.out.println("Piii"); }
}
```



Dekompozycja



```
class Silnik {
    SwiecaZaplonowa swieca;

    void zapal() {
        swieca.rozgrzej();
        swieca.dajIskre();
    }

    void odetnijDoplywPaliwa() { System.out.println("Psss"); }
    void wiecejGazu() { System.out.println("Brum"); }
    void mniejGazu() { System.out.println("Piii"); }
}
```

Quiz!

```
class Car {  
    String color;  
    /* ... */  
}
```

```
public class DontTrustYourWife {  
    public static void main(String[] args) {  
        Car myCar = new Car();  
        myCar.color = "black";  
        Car wifeCar = myCar;  
        wifeCar.color = "pink";  
        System.out.println(myCar.color);  
    }  
}
```




```
Car myCar = new Car();  
myCar.color = "black";
```





Car wifeCar = myCar;



```
wifeCar.color = "pink";
```




```
System.out.println(myCar.color);
```





```
Car myCar = new Car();  
myCar.color = "black";
```

```
Car wifeCar = new Car();  
wifeCar.color = "pink";
```



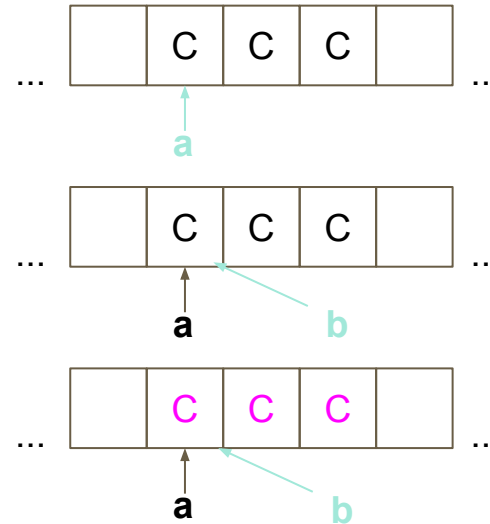
Referencje

```
Car a = new Car();
```

```
a.color = "black";
```

```
Car b = a;
```

```
b.color = "pink";
```



W zmiennych typów złożonych przypisujemy **referencje**.

Pakiety

Klasa zdefiniowana w katalogu głównym źródeł projektu (src) -
 pakiet domyślny

src/Lotto.java

```
public class Lotto {  
  
}
```

Nazwa klasy i pliku muszą się zgadzać!



Pakiety

Klasa zdefiniowana w pakiecie `pl.edu.agh.lottery`.

`src/pl/edu/agh/lottery/Lotto.java`

```
package pl.edu.agh.lottery;  
  
public class Lotto {  
  
}
```



Metody

```
class Calculator {  
    public int add(int a, int b) {  
        return a + b;  
    }  
  
    public boolean isNegative(int a) {  
        return a < 0;  
    }  
}
```

```
Calculator c = new Calculator();  
int num1 = 10;  
int num2 = -10;  
int sum = c.add(num1, 19);  
if (c.isNegative(num2)) {  
    // ...  
}
```

Metody

```
class Calculator {
    public int add(int a, int b) {
        return a + b;
    }
    public boolean isNegative(int a) {
        return a < 0;
    }
}
```

Diagram illustrating the components of the `add` method signature:

- sygnatura** (signature): Points to the entire method declaration `public int add(int a, int b)`.
- typ zwracany** (return type): Points to the `int` type.

```
Calculator c = new Calculator();
int num1 = 10;
int num2 = -10;
int sum = c.add(num1, 19);
if (c.isNegative(num2)) {
    // ...
}
```

"publiczna metoda o nazwie add, przyjmująca dwa parametry typu int o nazwach a i b, zwracająca int"

Metody

```
class Calculator {
    public int add(int a, int b) {
        return a + b;
    }

    public boolean isNegative(int a) {
        return a < 0;
    }
}
```

parametry

```
Calculator c = new Calculator();
int num1 = 10;
int num2 = -10;
int sum = c.add(num1, 19);
if (c.isNegative(num2)) {
    // ...
}
```

argumenty

Konstruktor = inicjalizacja obiektu

```
Car mojeAuto = new Car();
```

```
class Car {  
}
```

==

```
class Car {  
    public Car() {  
    }  
}
```

Pola klasy = stan obiektu

```
class Car {  
    private String color;  
  
    public Car(String color) {  
        this.color = color;  
    }  
  
    public String getColor() {  
        return this.color;  
    }  
}
```

Modyfikatory dostępu

```
class Car {
    private String color;

    public Car(String color) {
        this.color = color;
    }

    public String getColor() {
        return this.color;
    }
}
```

```
Car autoZony = new Car("pink");
```


```
String kolorAutoZony = autoZony.getColor();
```



```
String kolorAutoZony2 = autoZony.color;
```



```
autoZony.color = "black";
```



this

- odnosi się do instancji klasy, na rzecz której aktualnie wykonywany jest kod
- można go pominąć przy dostępie do pól lub metod, o ile nie zostały one przysłonięte przez zmienne lokalne lub parametry

```
class Car {  
    private String color;  
  
    public Car(String color) {  
        this.color = color;  
    }  
  
    public String getColor() {  
        return this.color;  
    }  
  
    public void setColor(String newColor) {  
        color = newColor;  
    }  
}
```



Przykładowa klasa

```
class Car {
    private String color;

    public Car(String color) {
        this.color = color;
    }

    public String getColor() {
        return this.color;
    }

    public void setColor(String color) {
        this.color = color;
    }
}
```

```
Car ferrari = new Car("red");
```

```
String color = ferrari.getColor(); ✓
```

```
ferrari.setColor("black"); ✓
```

```
ferrari.getColor(); ✓
```

```
// void result = ferrari.setColor("pink");
```



Przeciążanie metod

- pozwala na zdefiniowanie wartości domyślnych parametrów
- przeciążone metody mają taką samą nazwę, ale muszą różnić się listą argumentów (typami bądź ich liczbą)
- NIE wystarczy sama zmiana typu zwracanego

```
car.fill();
car.fill(34.6);
```

QUIZ! Czego brakuje w metodzie `fill(double howMuch)`?

```
class Car {
    private double tankCapacity;
    private double fuel;

    public double fill(double howMuch) {
        fuel += howMuch;
        return tankCapacity - fuel;
    }

    public double fill() {
        return fill(tankCapacity - fuel);
    }

    //public boolean fill(double howMuch) {
    //    fuel += howMuch;
    //    return tankCapacity <= fuel;
    //}
}
```

Co można stworzyć w Javie?

aplikacje

Java jako programy na komputery i urządzenia mobilne

~~applety~~

~~Java jako programy uruchamiane w przeglądarce internetowej~~

servlety

Java jako programy uruchamiane na serwerach, pełniące funkcje serwera WWW (często w postaci Java Server Pages (JSP))

Edycje Javy

Java SE (Standard Edition)

Java EE (Enterprise Edition)

Java ME (Micro Edition)

Aplikacje i applety

Servlety

Aplikacje na urządzenia o małych zasobach
(kiedyś telefony komórkowe, dziś systemy wbudowane)

Java API

- Z każdą wersją Javy wydawane jest także JDK
- Oprócz narzędzi (np. `java`, `javac`), JDK zawiera Java Application Programming Interface (API) znane też jako *Standard library*
- Gotowe klasy i interfejsy, które dostarczają funkcjonalności do wykorzystania w aplikacjach, które tworzymy



Java API - historia

JDK	Wydanie	Liczba klas
Java SE 12 / JDK 12	2019	4433
Java SE 9 / JDK 1.9.0	2017	6005
Java SE 8 / JDK 1.8.0	2014	4240
Java SE 7 / JDK 1.7.0	2011	4024
Java 2 SE 5.0 / JDK 1.5.0	2004	3279
Java 2 SE / SDK 1.4.0	2002	2991
Java 2 SE / SDK 1.3	2000	1842
Development Kit 1.0	1996	212

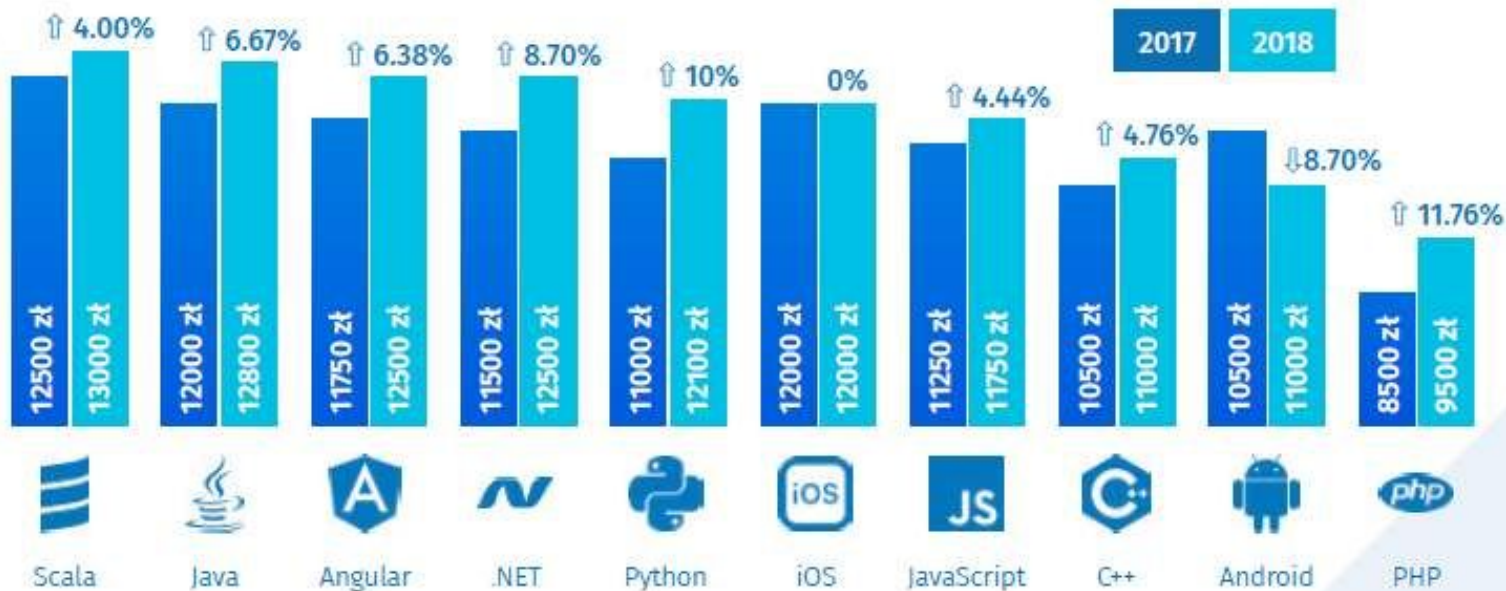
Java API - przykłady

<code>java.lang</code>	Podstawowe elementy języka (np. proste typy danych np. String)
<code>java.util</code>	Złożone typy danych (np. kolekcje), operacje na datach, generator liczb losowych
<code>java.io</code>	Operacje Input/Output (I/O) - np. operacje na plikach
<code>java.math</code>	Precyzyjne obliczenia arytmetyczne (np. <code>BigDecimal</code>)
<code>java.net</code>	Komunikacja sieciowa

```
import java.util.Random;
import java.util.Arrays;
public class Lotto {
    public static void main(String[] args) {
        int[] lottery = new int[6];
        Random random = new Random();
        for (int draw = 0; draw < 6; draw++) {
            lottery[draw] = random.nextInt(49) + 1;
        }
        Arrays.sort(lottery);
        System.out.println(Arrays.toString(lottery));
    }
}
```



Marchewka



źródło: No Fluff Jobs,

<https://forsal.pl/artykuly/1388649,zarobki-programistow-2019-jezyki-programowania-kontra-wynagrodzenia.html>

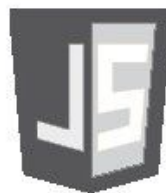
Java Development Tools

- Można pisać w notatniku, ale
- IDE (Integrated Development Environment) - narzędzie wspierające tworzenie oprogramowania
- Przykładowe IDE dla Javy:
 - IntelliJ
 - Eclipse
 - NetBeans



Czym Java NIE jest?

JAVA *is to* JAVASCRIPT *as* HAM *is to* HAMSTER



Literatura



- *Head First Java* - Kathy Sierra & Bert Bates
- *Effective Java* - Joshua Bloch
- *Java Puzzlers* - Joshua Bloch
- *Java Concurrency in Practice* - Brian Goetz
- *Thinking in Java* - Bruce Eckel
- Java Language Specification (JLS)

<https://docs.oracle.com/javase/specs/>



Żargon



Quiz!

```
public class GoodByeJava {  
    public static void main(String[] args) {  
        Integer a = 200;  
        Integer b = 200;  
        System.out.println(a == b);  
    }  
}
```

