

## Przewodnik 4 – praca z HashMap'ą i HashSet'em.

dr inż. Łukasz Sosnowski  
WIT Wyższa Szkoła Informatyki Stosowanej i Zarządzania  
pod auspicjami Polskiej Akademii Nauk

## 1 Utworzenie klasy

W pakiecie **pl.wit.lab2** utwórz klasę o nazwie: **Lab2SetAndMapExample**. Dodaj komentarz wieloliniowy dla klasy: „Klasa z przykładami metod operującymi na HashMapie i HashSet”

## 2 Zadeklarowanie zmiennych składowych klasy

Zadeklaruj w klasie zmienną prywatną o nazwie „setCars” która będzie typu Set<String>:

```
private Set<String> setCars=null;
```

Zadeklaruj drugą zmienną prywatną o nazwie „mapCarPower”, która będzie następującą mapą Map<String,Integer>:

```
private Map<String,Integer> mapCarPower=null;
```

Pierwsza zmienna stanowi zbiór marek samochodów, druga mapę mocy samochodów.

Kluczem mapy jest łańcuch znaków a wartością wartość całkowita.

Dla wyżej zadeklarowanych zmiennych wygeneruj automatycznie gettery (Source → Generate Getters and Setters...).

Ponadto powołaj w klasie zmienną loggera:

```
protected static final Logger log =  
Logger.getLogger(Lab2SetAndMapExample.class.getName());
```

## 3 Konstruktor

Zdefiniuj konstruktor publiczny bezparametryczny. Zaimplementuj go w taki sposób aby zainicjować obie zmienne odpowiednio pustym zbiorem oraz pustą mapą;

```
public Lab2SetAndMapExample () {
```

```
setCars = new HashSet<String>();  
mapCarPower = new HashMap<String,Integer>();  
}
```

## 4 Metody klasy

a. Zaimplementuj metodę publiczną bezparametrową *printSetToLog* nie zwracającą wartości, która będzie wypisywać do logu w sposób bezpieczny w trybie „info” zawartość zmiennej typu Set.

Do implementacji użyj instrukcji „if” do sprawdzenia oraz pętli rozszerzonej „for”.

```
if(setCars!=null) {  
    for(String el:setCars)  
        Log.info(""+el+", ");  
}
```

b. Zaimplementuj metodę publiczną bezparametrową *printMapToLog* nie zwracającą wartości, która będzie wypisywać do logu w sposób bezpieczny w trybie „info” zawartość zmiennej typu Map.

Do implementacji użyj instrukcji „if” do sprawdzenia oraz pętli rozszerzonej „for”.

```
if(mapCarPower!=null) {  
    for(String el:mapCarPower.keySet())  
        Log.info(""+el+"="+mapCarPower.get(el)+", ");  
}
```

c. Zaimplementuj 2 metody publiczne nie zwracające wartości o nazwie *addElement* przyjmująca odpowiednio parametry: *String* oraz *String i Integer*.

Dla pierwszej metody kod może wyglądać następująco:

```
public void addElement(String element) {  
    if(setCars!=null)  
        setCars.add(element);  
}
```

Dla drugiej metod kod może wyglądać następująco:

```
public void addElement(String key, Integer value) {  
    if(mapCarPower!=null)  
        mapCarPower.put(key,value);  
}
```

d. Zaimplementuj 2 metody publiczne jednoparametrowe o nazwach:  
*getMapValue(String key)*, *isSetContains(String element)*

zwracające odpowiednio wartości *Integer* i *boolean*;

Deklaracje metod będą następujące:

```
public Integer getMapValue(String key) {  
    //Implementacja  
}
```



```
public boolean isSetContains(String element){  
}
```

Wykonaj ich implementację.

## 5 Test jednostkowy

W pakiecie **pl.wit.lab2** katalogu `src/test/java` utwórz klasę testu jednostkowego o nazwie: **Lab2SetAndMapExampleTest**.

Zaimplementuj metody testowe dla wykonanych metod w pkt. 4