Java Anonymous Classes

Τι είναι anonymous class?

- Anonymous class είναι μια ανώνυμη κλάση, χωρίς όνομα, για την οποία δημιουργούμε και αντικείμενο αμέσως με τη δήλωσή της
- Δεν χρησιμοποιούνται για τη δημιουργία πολλαπλών αντικειμένων
- Χρησιμοποιούνται για τη δήλωση και την αρχικοποίηση ενός αντικειμένου μέσα σε ένα expression
- Μπορούν να οριστούν με 2 (+1) τρόπους:
  - Extending an existing class
  - Implementing an existing interface
  - Με τον «κλασικό» τρόπο δημιουργίας στιγμιότυπου, χωρίς την ύπαρξη μεταβλητής. Έτσι βέβαια δεν θα δημιουργηθεί νέα κλάση, αλλά μόνο ένα ανώνυμο αντικείμενο. Συμπερασματικά, anonymous class έχουμε με 2 τρόπους, ενώ για anonymous object μπορούμε να χρησιμοποιήσουμε ακόμα έναν τρόπο

Extending an existing class

Implementing an existing interface

Using new without a variable

- Ο 3ος τρόπος είναι ουσιαστικά να χρησιμοποιηθεί ένα αντικείμενο απλώς χωρίς την ύπαρξη μεταβλητής, μόνο με τη χρήση του στιγμιοτύπου
- new Object()
- Εδώ βέβαια δεν δημιουργείται νέα τάξη, μόνο ένα νέο αντικείμενο
- Αν γίνει τροποποίηση οποιουδήποτε χαρακτηριστικού του νέου αντικειμένου κατά το instantiation, τότε θα ισχύει η πρώτη περίπτωση -> **extending an existing class**

Παραδείγματα

```java
package com.unipi.talepis;

public class Human {
    private int age;
    private String name;
    public void eat(){
        System.out.println("I am eating!");
    }
}
```

```java
package com.unipi.talepis;

public interface ISee {
    void see(String s);
}
```

Extending an existing Class

```
package com.unipi.talepis;

public class Main {

    public static void main(String[] args) {
        new Human(){
            @Override
            public void eat() {
                System.out.println("Hello, I am eating now!");
            }
        }.eat();
    }
}
```

```
C:\Java\jdk-9.0.4\bin\java.exe "-javaagent:
Hello, I am eating now!

Process finished with exit code 0
```

Implementing an existing Interface

```
package com.unipi.talepis;

public class Main {

    public static void main(String[] args) {
        new ISee(){
            @Override
            public void see(String s) {
                System.out.println("I am watcing "+s);
            }
        }.see( s: "Unipi");
    }
}
```

```
C:\Java\jdk-9.0.4\bin\java.exe "-javaagent:
I am watcing Unipi

Process finished with exit code 0
```

Με τη χρήση απλού “new”

```
package com.unipi.talepis;

public class Main {

    public static void main(String[] args) {
        new Human().eat();
    }
}
```

```
C:\Java\jdk-9.0.4\bin\java.exe "-javaagent:
I am eating!

Process finished with exit code 0
```

- Για τη χρήση ενός απλού anonymous object μπορεί να χρησιμοποιηθεί και οποιαδήποτε συνάρτηση επιστρέφει το εν λόγω object

```
package com.unipi.talepis;

public class Human {
    private int age;
    private String name;
    public void eat(){
        System.out.println("I am eating!");
    }
    public static Human getAHuman(){
        return new Human();
    }
}
```

```java
package com.unipi.talepis;

public class Main {

    public static void main(String[] args) {
        //new Human().eat();
        Human.getAHuman().eat();
    }
}
```

```
C:\Java\jdk-9.0.4\bin\java.exe "-javaagent:
I am eating!

Process finished with exit code 0
```

Χρήση μεταβλητών από anonymous classes

- Μια ανώνυμη τάξη έχει πλήρη πρόσβαση σε member variables της τάξης στην οποία ανήκει (οι anonymous classes θα είναι «αναγκαστικά» εσωτερικές τάξεις)
- Μια ανώνυμη τάξη ΔΕΝ μπορεί να έχει πρόσβαση σε τοπικές μεταβλητές (εντός μεθόδων), οι οποίες δεν είναι δηλωμένες ως final, ή έστω να είναι ουσιαστικά τελικές (effectively final)

Παραδείγματα

```
public static void main(String[] args) {
    /*new ISee(){
        @Override
        public void see(String s) {
            System.out.println("I am watching "+s);
        }
    }.see("Unipi");*/
    String temp = "John";
    final String t = "Jim";
    new Human(){
        int i = 5;
        String s = args[0];
        String g = t;
        //String v = temp; //will compile if it is effective final
        private void go(){
            //temp = "Bob"; //problem
            System.out.println(s);
        }
    }.go();
}
```

Anonymous class with no default constructor

```
public class Student extends Human{
    private int AM;
    protected String email;
    public String department;

    public Student(int AM) {
        this.AM = AM;
    }
    public void speak(){
        System.out.println("My am is "+AM);
    }
}
```

```
new Student( AM: 432){
}.speak();
```

Anonymous class with instance initializer

```
public class Student extends Human{
    private int AM;
    protected String email;
    public String department;

    public Student(int AM) {
        this.AM = AM;
    }
    public void speak(){
        System.out.println("My am is "+AM);
    }
}
```

```
new Student( AM: 53){
    {
        department="Informatics";
        email = "me@unipi.gr"; //why?...
        //AM = 54; //of course not...
    }
}.speak();
```

Τι μπορούμε να δηλώσουμε εντός μιας anonymous class

- Fields (instance, static need to be final and initialized)
- Extra methods (even if they do not implement any methods of the supertype)
- Instance initializers
- Local classes (εσωτερικές τάξεις)
- ΔΕΝ επιτρέπεται εσωτερικό interface

Χρήση anonymous object

- Η χρήση των anonymous class/object δεν γίνεται συνήθως όπως στα προηγούμενα παραδείγματα (τα οποία δημιουργήθηκαν για εκπαιδευτικούς λόγους), αλλά μέσω παραμέτρων σε μεθόδους
- Οι μέθοδοι αυτοί επιβάλλουν χρήση:
  - είτε αντικειμένων που είναι υποτάξεις μια συγκεκριμένης τάξης
  - είτε αντικειμένων που υλοποιούν κάποιο συγκεκριμένο Interface
- Επίσης, εξίσου πολλές φορές τα anonymous objects αποτελούν μέρος ενός expression απόδοσης instance σε μια μεταβλητή