תכנות מכוון עצמים ושפת JAVA

הרצאה 05

תכונות ושיטות סטטיות

ביחידה זו נלמד:

- תכונות סטטיות
- שיטות סטטיות
- שימוש בתכונות סטטיות כקבועים
- המחלקה java.lang.Math
  - המחלקה java.util.Random
- המחלקה Arrays
- enum

תכונות סטטיות

- תכונת מופע (Instance Attribute)
  - עד כה ראינו שתכונה במחלקה משוכפלת עבור כל אובייקט הנוצר מהמחלקה
- תכונת מחלקה (תכונה סטטית)
  - תכונה שיש עותק אחד שלה עבור כל האובייקטים מהמחלקה
  - כל האובייקטים מאותה מחלקה יכולים לקרוא ולשנות תכונה זו
  - למשל עבור תכונות שלא נרצה שיהיו שונות בין כל האובייקטים, אלא יהיו עם ערך זהה
  - תכונה סטטית קיימת עוד לפני שנוצר אפילו אובייקט אחד מהמחלקה

דוגמא: Person

משתנה סטטי, מאותחל ל- 0

```
Name:   Gogo      Age:   14 (can not drive)
Name:   Momo      Age:   23 (can drive)
Name:   Yoyo      Age:   19 (can drive)
Changing adult age to be 21:
Name:   Gogo      Age:   14 (can not drive)
Name:   Momo      Age:   23 (can drive)
Name:   Yoyo      Age:   19 (can not drive)
```

```
public class Person   {
   private static int licenseAge;
   private string name;
   private int age;

   public Person(string name, int age)  {
      this.name = name;
      this.age = age;
   }

   public void setLicenseAge(int age)  {
      licenseAge = age;
    }

   public string toString()  {
      string str = "";
      str += "Name:  " + name;
      str += "\tAge:  " + age + " (";
      if (age < licenseAge)
         str += “can not drive";
      else
         str += “can drive";
      str += ")";
      return str;
   }
} // class Person
```

Person::licenseAge=21

licenseAge =
name=“Momo”
age=23

licenseAge =
name=“Yoyo”
age=19

שימוש במשתנה הסטטי

licenseAge =
name=“Gogo”
age=14

```
public static void main(String[] args)  {
   Person p1 = new Person("Gogo", 14);
   Person p2 = new Person("Momo", 23);
   Person p3 = new Person("Yoyo", 19);
   p1. setLicenseAge(18); // same as: p2. setLicenseAge (18

   System.out.println(p1.toString());
   System.out.println(p2.toString());
   System.out.println(p3.toString());

   System.out.println("Changing adult age to be 21:");
   p2.setLicenseAge(21); // same as: p3.setLicenseAge (21)
   System.out.println(p1.toString());
   System.out.println(p2.toString());
   System.out.println(p3.toString());
} // main
```

משתנה סטטי כקבוע במחלקה

- יתכן ונרצה שהמשתנה יהיה קבוע, משמע שלא ניתן לשנותו
- קבוע זה יהיה משותף לכל האובייקטים מטיפוס המחלקה ולכן נרצה שהוא יהיה חלק מהמחלקה (למשל ADULT_AGE)
  - מאחר וקבוע זה משותף לכל האובייקטים עליו להיות static
  - מאחר והוא קבוע ולא נרצה שישנו אותו נגדיר אותו כ- final
  - מאחר ולא ניתן לשנות את ערכו ניתן להגדיר קבוע זה כ- public
  - מאחר ומשתנה סטטי נוצר לפני יצירת אפילו אובייקט אחד, והוא public ניתן לגשת אליו רק עם שם המחלקה
- מקובל להגדיר קבועים באותיות גדולות (ראו המלצה זו כמחייבת!)

דוגמא למשתנה סטטי כקבוע במחלקה

```
public class Person {

    public static final int ADULT_AGE = 18;

    private String name;
    private int age;

    public Person(String name, int age) {
        this.name = name;
        this.age = age;
    }

    public String toString() {
        String str = "";
        str += "Name:   " + name;
        str += "\tAge:   " + age + " (";
        if (age < ADULT_AGE)
            str += "child";
        else
            str += "adult";
        str += ")";
        return str;
    }
} // class Person
```

```
public static void main(String[] args)
{
    System.out.println("Adult age is: "
                       + Person.ADULT_AGE);

    Person p1 = new Person("Gogo", 14);
    Person p2 = new Person("Momo", 23);
    Person p3 = new Person("Yoyo", 19);

    System.out.println(p1.toString());
    System.out.println(p2.toString());
    System.out.println(p3.toString());
} // main
```

```
Adult age is: 18
Name:   Gogo      Age:   14 (child)
Name:   Momo      Age:   23 (adult)
Name:   Yoyo      Age:   19 (adult)
```

יצירת ID אוטומטי

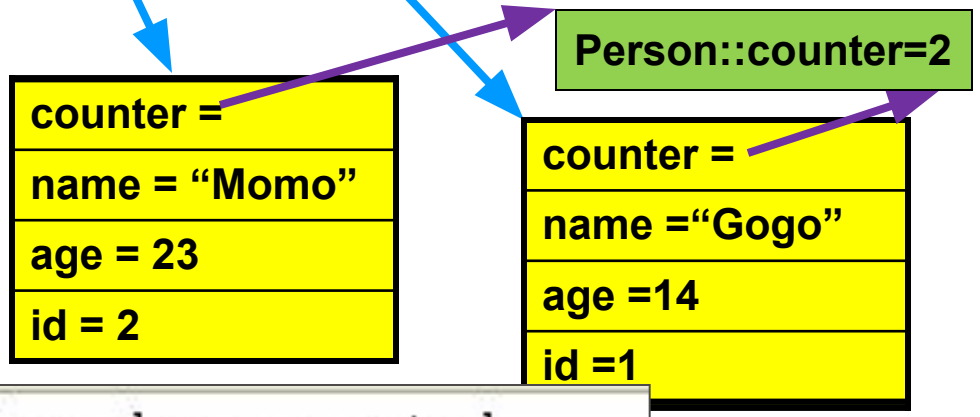
```
public class Person   {
    private static int counter;
    private string name;
    private int age;
    private int id;

    public Person(string name, int age)  {
        this.name = name;
        this.age = age;
        id = ++counter;
    }

    public int getNumOfPersons()    {
        return counter;
    }

    public string toString()    {
        string str = "";
        str += "Id:  " + id;
        str += "\tName:  " + name;
        str += "\tAge:  " + age;
        return str;
    }
} // class Person
```

```
static void main(String[] args)    {
    Person p1 = new Person("Gogo", 14);
    System.out.println(   p1.getNumOfPersons()
                    +  " persons have been created");
    Person p2 = new Person("Momo", 23);
    System.out.println(   p1.getNumOfPersons()
                    + " persons have been created");
    System.out.println(p1.toString());
    System.out.println(p2.toString());
    System.out.println(p2.getNumOfPersons()
                    + " persons have been created");
} // main
```




```
1 persons have been created
2 persons have been created
Id:   1   Name:   Gogo      Age:   14
Id:   2   Name:   Momo      Age:   23
2 persons have been created
```

שיטות סטטיות

- שיטה סטטית היא שיטה הנכתבת בתוך מחלקה, אך אין צורך לייצר אובייקט על מנת להפעיל אותה
- נכתוב שיטה כסטטית במקרה בו אינה מתבססת על נתוניו של אובייקט מסוים, אך קשורה לוגית למחלקה
  - שיטה סטטית יכולה לגשת למשתנים סטטיים, אך לא למשתנים רגילים (משתני מופע)
  - שיטה רגילה יכולה לגשת למשתנים סטטיים
- קריאה לשיטה מתבצעת באמצעות שם המחלקה
  - היתרון: ניתן לקרוא לשיטה עוד לפני שנוצר אפילו אובייקט אחד

דוגמא: החזרת מספר האנשים שנוצרו

```
public class Person {
    private static int counter;
    private String name;
    private int age;
    private int id;

    public Person(String name, int age) {
        this.name = name;
        this.age = age;
        id = ++counter;
    }

    public static int getNumOfPersons() {
        return counter;
    }

    public String toString() {
        return "Id: " + id + "\tName:   " +
            name + "\tAge:   " + age;
    }

} // class Person
```

שיטה סטטית

```
public static void main(Strin
{
    System.out.println(Person.getNumOfPersons() +
                       " persons have been created");
    Person p1 = new Person("Gogo", 14);
    Person p2 = new Person("Momo", 23);
    System.out.println(Person.getNumOfPersons() +
                       " persons have been created");
    System.out.println(p1.getNumOfPersons() + " " +
            "              pers    have been created");

    System.out.println(p1.toS
    System.out.println(p2.toS
} // main
```

קריאה לשיטה הסטטית בעזרת שם המחלקה

קריאה לשיטה הסטטית בעזרת אובייקט

```
0 persons have been created
2 persons have been created
2 persons have been created
Id: 1    Name:   Gogo      Age:   14
Id: 2    Name:   Momo      Age:   23
```

מחלקות הנותנות שירותים

- ישנן מחלקות שרק נותנות שירותים, כלומר יש להן אוסף שיטות ללא תכונות
- במחלקה כזו כל השיטות הן סטטיות, שכן אם אין תכונות, אין משמעות לאובייקט
- יתכן ומחלקה זו גם תכיל אוסף של קבועים
- למשל, המחלקה Math

המחלקה Math

- המחלקה Math מכילה שיטות מתמטיות, שכולן סטטיות וכן משתנים סטטיים
- דוגמאות:
  - הקבועים E ו- PI
  - השיטות:
    - Abs
    - Cos
    - Pow
    - Sqrt
    - ועוד רבות, מומלץ להסתכל!

המחלקה Math – דוגמת שימוש

```
public static void main(String[] args)
{
    System.out.println("2^3=" + Math.pow(2, 3));

    System.out.println("Perimeter of a circle with radius 3 is " + 2 * 3*Math.PI);

    System.out.println("Sqrt of 9 is " + Math.sqrt(9));

} // main
```

```
2^3=8.0
Perimeter of a circle with radius 3 is 18.84955592153876
Sqrt of 9 is 3.0
```

קבלת מספרים אקראיים

```
2  public class Program {
3
4      public static void main(String[] args) {
5          for (int i=0 ; i < 20 ; i++)
6              System.out.println(Math.random());
7      }
8  }
```

```
0.1894182295556564 6
0.1728294742334301
0.0018494963068560466
0.11678076799876602
0.526257351555487 5
0.07530436715011724
0.6256380233276815
0.727932709397 62
0.04968085172849157
0.8831279170276081
0.4591833595365548 7
0.5423195609804661
```

השיטה random מחזירה
מספר עשרוני בטווח 0-1

קבלת מספרים אקראיים בטווח מסויים

- נכפיל את הערך המוחזר בכמות המספרים שנרצה בטווח, ונעשה casting ל- int

```
4      public static void main(String[] args) {
5          System.out.println("Random between 0-2:");
6          for (int i=0 ; i < 5 ; i++)
7              System.out.print((int)(Math.random()*3) + " ");
8
9          System.out.println("\nRandom between 0-9:");
10         for (int i=0 ; i < 5 ; i++)
11             System.out.print((int)(Math.random()*10) + " ");
12
13         System.out.println("\nRandom between 1-10:");
14         for (int i=0 ; i < 5 ; i++)
15             System.out.print((int)(Math.random()*10) + 1 + " ");
16     }
```

```
Random between 0-2:
0 1 2 2 0
Random between 0-9:
4 6 3 4 5
Random between 1-10:
1 3 6 10 1
```

המחלקה Random

```
import java.util.Random;
```

אובייקט מהמחלקה Random מגריל ערכים בהתפלגות נורמלית

```
 6      public static void main(String[] args) {
 7          Random r = new Random();
 8
 9          System.out.println("Some random numbers:");
10          for (int i=0 ; i < 5 ; i++)
11              System.out.print(r.nextInt() + " ");
12
13          System.out.println("\nSome random numbers from 0-9:");
14          for (int i=0 ; i < 5 ; i++)
15              System.out.print(r.nextInt(10) + " ");
16
17          System.out.println("\nSome random numbers floats between 0-1:");
18          for (int i=0 ; i < 5 ; i++)
19              System.out.print(r.nextFloat() + " ");
20      }
```

```
Some random numbers:
969146129 -227713963 540858681 -1530789359 1855799536
Some random numbers from 0-9:
4 2 1 5 4
Some random numbers floats between 0-1:
0.79736644 0.70451397 0.16508561 0.78559214 0.5804044
```

המחלקה Arrays

Arrays.copyOf היא שיטה סטטית המקבלת כפרמטר מערך להעתקה ואת הגודל של המערך החדש. השיטה מבצעת השמה בין איבר לאיבר. אם גודל המערך החדש יותר גדול, שאר איבריו יאופסו.

Arrays.equals היא שיטה סטטית המקבלת 2 מערכים ובודקת האם הם שווים: האם אורכם זהה והאם האיברים במקומות המתאימים זהים

```
3  public class Program {
4
5      public static void main(Stri[] args) {
6          int[] arr = {2,3,7,1,5};
7          int[] newArr = Arrays.copyOf(arr, 10);
8
9          System.out.println("arr is    : " + Arrays.toString(arr));
10         System.out.println("newArr is: " + Arrays.toString(newArr));
11         System.out.println(Arrays.equals(arr, newArr));
12
13         int[] newArr2 = Arrays.copyOf(arr, arr.length);
14         System.out.println("newArr2 is: " + Arrays.toString(newArr2));
15         System.out.println(Arrays.equals(arr, newArr2));
16     }
```

```
arr is    : [2, 3, 7, 1, 5]
newArr is: [2, 3, 7, 1, 5, 0, 0, 0, 0, 0]
false
newArr2 is: [2, 3, 7, 1, 5]
true
```

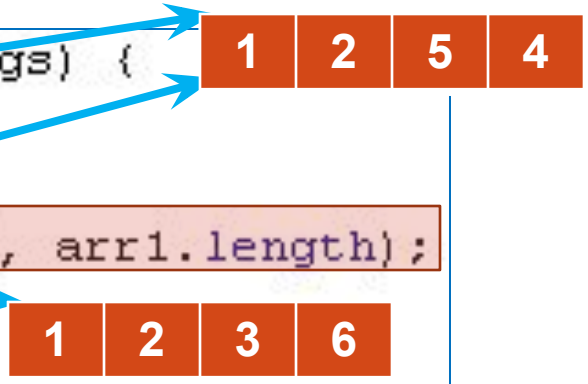
Arrays.toString היא שיטה סטטית המקבלת כפרמטר מערך ומחזירה מחרוזת כך שאיברי המערך מופרדים ע"י פסיק

אבחנה בין העתקת מערך לבין העתקת הפניה

```
public static void main(String[] args) {
    int[] arr1 = {1,2,3,4};

    int[] arr2 = Arrays.copyOf(arr1, arr1.length);
    int[] arr3 = arr1;

    arr3[2] = 5;
    arr2[3] = 6;
}
```



כלומר, העתקת מערך יוצרת מופע נוסף בלתי תלוי של האובייקט, בעוד שהשמה רק משנה את ההפניה.

כיצד עובדת השיטה Arrays.copyOf עבור אובייקטים?

- האם copyOf מייצרת העתקים של האובייקטים או רק מפנה אליהם?

```
public static void main(String[] args) {
    Point[] points = new Point[3];
    for (int i=0 ; i < points.length ; i++)
        points[i] = new Point(i, i);

    Point[] copyPoints = Arrays.copyOf(points, points.length);

    points[0].setX(77);
    System.out.println("points:      " + Arrays.toString(points));
    System.out.println("copyPoints:  " + Arrays.toString(copyPoints));
}
```

```
points:      [(77, 0), (1, 1), (2, 2)]
copyPoints:  [(77, 0), (1, 1), (2, 2)]
```

מאחר ושינוי במערך אחד גרר שינוי במערך השני, משמע השיטה אינה מייצרת העתקים לאובייקטים, אלא רק מבצעת השמה.

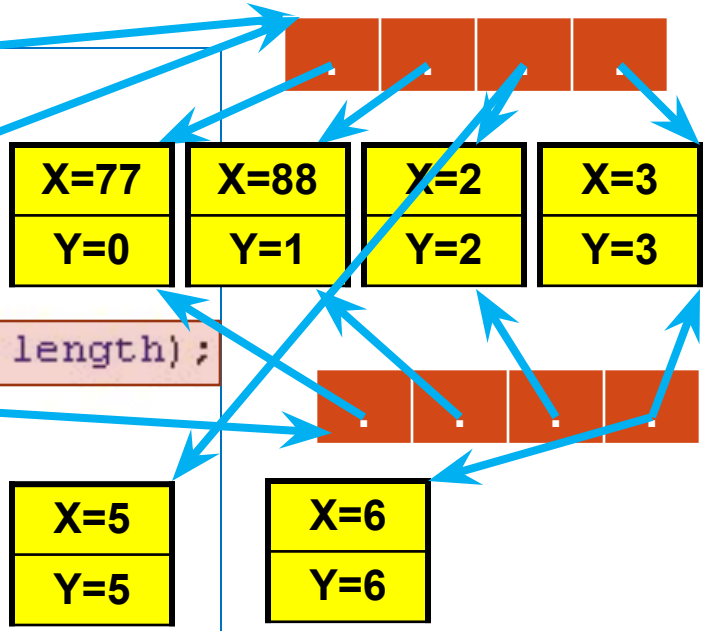
Arrays ואובייקטים – דוגמא נוספת

```
public static void main(String[] args) {
    Point[] arr1 = new Point[4];
    for (int i=0 ; i < arr1.length ; i++)
        arr1[i] = new Point(i, i);

    Point[] arr2 = Arrays.copyOf(arr1, arr1.length);
    Point[] arr3 = arr1;

    arr3[0].setX(77);
    arr2[1].setX(88);

    arr3[2] = new Point(5, 5);
    arr2[3] = new Point(6, 6);
}
```



עכשיו:

- **ל- arr1 ו- arr3 יש הפניה יחודית לאיבר באינדקס 2.**
- **ל- arr2 יש הפניה יחודית לאיבר באינדקס 3.**

מהו enum

- enum הינה דרך להגדרת טיפוס חדש שערכיו יהיו מקבוצת קבועים בעלי קשר לוגי שיוגדרו עבורו
- קבועים אלו יהיו מספרים סידורים החל מ- 0
- ניתן להגדיר משתנה מטיפוס קבוצה זו וערכו יהיה רק מקבוצת הקבועים שהוגדרו בקבוצה

דוגמא

```
3  public class Program {
4
5      enum Desert {IceCream, Cake, Cookies};
6
7      public static void main(String[] args) {
8          Scanner s  = new Scanner(System.in);
9
10         Desert d = Desert.IceCream;
11
12         System.out.println(d.ordinal() + " --> " + d.name());
13
14         System.out.println("\nAll values:");
15         Desert[] allDeserts = Desert.values();
16         for (int i=0 ; i < allDeserts.length ; i++)
17             System.out.println(allDeserts[i].ordinal() + " --> " + allDeserts[i].name());
18
19         System.out.println("\nWhat is you favorite desert? ");
20         Desert ans = Desert.valueOf(s.next());
21         System.out.println("Your favorite desert is " + ans + " (" + ans.ordinal() + ")");
22     }
23 }
```

הגדרת קבוצת קבועים ע"י enum

מתן ערך למשתנה מטיפוס הקבוצה

קבלת הערך המספרי של הקבוע

קבלת השם של הקבוע

קבלת מערך עם כל איברי הקבוצה

קליטת ערך למשתנה באמצעות קבלת שם הקבוע.
אם יוכנס ערך שאינו בקבוצה תיזרק חריגה.

```
0 --> IceCream

All values:
0 --> IceCream
1 --> Cake
2 --> Cookies

What is you favorite desert?
Cake
Your favorite desert is Cake (1)
```

הגדרת enum בתוך מחלקה

```
1  public class Food {
2      enum Desert {IceCream, Cake, Cookies};
3      enum JunkFood {Pizza, Hamburger, Falafel};
4
5      private Desert theDesert;
6      private JunkFood theJunk;
7
8      public Food(Desert theDesert, JunkFood theJunk) {
9          this.theDesert = theDesert;
10         this.theJunk = theJunk;
11     }
12
13     @Override
14     public String toString() {
15         return "Desert=" + theDesert + ", Junk=" + theJunk;
16     }
17 }
```

שימוש בקבוע enum שהוגדר בתוך מחלקה יהיה עם קידומת שם המחלקה

```
public static void main(String[] args) {
    Food myFavouriteFood = new Food(Food.Desert.Cake, Food.JunkFood.Pizza);
    System.out.println("My favorite food is: " + myFavouriteFood.toString());
}
```

```
My favorite food is: Desert=Cake, Junk=Pizza
```

ביחידה זו למדנו:

- תכונות סטטיות
- שיטות סטטיות
- שימוש בתכונות סטטיות כקבועים
- המחלקה java.lang.Math
  - המחלקה java.util.Random
- המחלקה Arrays
- enum