מחלקות פנימיות (מקוננות)
Inner (Nested) Classes

אוהד ברזילי

תכנות מתקדם בשפת Java

אוניברסיטת תל אביב

Inner Classes

- מחלקה פנימית היא מחלקה שהוגדרה בתחום (Scope – בין המסולסליים) של מחלקה אחרת
- דוגמא:

```
public class House {
    private String address;
    public class Room {
        private double width;
        private double height;
    }
}
```

שימוש לב!

Room אינה שדה של המחלקה House

מחלקות פנימיות

- הגדרת מחלקה כפנימית מרמזת על היחס בין המחלקה הפנימית והמחלקה העוטפת:
  - למחלקה הפנימית יש משמעות רק בהקשר של המחלקה החיצונית
  - למחלקה הפנימית יש הכרות אינטימית עם המחלקה החיצונית
  - המחלקה הפנימית היא מחלקת עזר של המחלקה החיצונית
- דוגמאות:
  - `Collection` ו- `Iterator`
  - `Body` ו- `Brain`

Inner Classes

- **ב Java כל מופע של עצם מטיפוס המחלקה הפנימית צריך להיות משויך לעצם מטיפוס המחלקה העוטפת**
- **השלכות**
  - תחביר מיוחד לבנאי
  - לעצם מטיפוס המחלקה הפנימית יש שדה הפנייה שמיוצר אוטומטית לעצם מהמחלקה העוטפת
  - כתוצאה לכך יש למחלה הפנימית גישה לשרותים (אפילו פרטיים!) של המחלקה העוטפת

Inner Classes

```
public class House {
    private String address;
    public class Room {
        // hidden reference to a House
        private double width;
        private double height;
        public String toString(){
            return  "Room inside: " +  address;
        }
    }
}
```

```
public class House {
    private String address;
    private double  height;
    public class Room {
        // hidden reference to a House
        private double height;
        public String toString(){
            return  "Room height: " + height
            + "  House height:" + House.this.height;
        }
    }
}
```

Height of *House*

Height of *Room*

Height of *Room*
Same as *this.height*

יצירת מופעים

- כאשר המחלקה החיצונית יוצרת מופע של עצם מטיפוס המחלקה הפנימית אזי העצם יווצר בהקשר של העצם היוצר

- כאשר עצם מטיפוס המחלקה הפנימית נוצר מחוץ למחלקה העוטפת, יש צורך בתחביר מיוחד

יצירת מופע ע"י המחלקה החיצונית

```
public class House {
    private String address;
    public void test(){
        Room  r = new Room();
        System.out.println( r );
    }
    public class Room {
        …
    }
}
```

יצירת מופע שלא ע"י המחלקה החיצונית

```
public class Test {
    public static void main(String[] args){
        House  h = new House();
        House.Room  r =  h.new Room();
    }
}
```

```
outerObject.new InnerClassName
```

Static Nested Classes

- ניתן להגדיר מחלקה פנימית כ `static` ובכך לציין שהיא אינה קשורה למופע מסויים של המחלקה העוטפת

- הדבר אנלוגי למחלקה שכל שרותיה הוגדרו כ `static` והיא משמשת כמחלקת עזר (מחלקת שרות) עבור מחלקה מסוימת

- בשפת C++ יחס זה מושג ע"י הגדרת יחס `friend`

```
public class House {
    private String address;
    public static class Room {
        public String toString(){
            return  "Room " +  address;
        }
    }
}
```

Error: this room is not related to any house

```
public class Test {
    public static void main(String[] args){
        House.Room  r =  new House.Room();
        …
```

Not related to any house

```
new OuterClassName.InnerClassName
```

הגנה על מחלקות פנימיות סטאטיות

- אם המחלקה הפנימית אינה ציבורית (אינה מוגדרת public), הטיפוס שלה מוסתר, אבל עצמים מהמחלקה אינם מוסתרים אם יש התייחסות אליהם

```
public class Outer … {
    private static class Inner implement Inter {…}
    public static Inter getInner() {
        return new Inner ();
    }
     …

Inter i = new Outer.Inner(); //error
Inter i = Outer.getInner();   // ok
```

מחלקות מקומיות - מחלוקת פנימיות בתוך מתודות

- ניתן להגדיר מחלקה פנימית בתוך מתודה של המחלקה החיצונית
- הדבר מגביל את תחום ההכרה של אותה מחלקה לתחום אותה המתודה בלבד
- המחלקה הפנימית תוכל להשתמש במשתנים מקומיים של המתודה רק אם הם הוגדרו כ `final` (מדוע?)

```
public class Test {
    …
    public void test () {
        class Info {
            private int x;
            public Info(int x) {this.x=x;}
            public String toString() {return "** " + x + "**" ;}
        };
        Info  inf1 = new Info(0);
        System.out.println(inf1);
    }
}
```

שימוש במשתנים מקומיים

```
public class Test {
    public void test (int x) {
      final int y = x+3;
      class Info {
            public String toString(){ return "**" +  y + "***"; }
      };
      System.out.println( new Info());
    }
}
```

מחלקות אנונימיות

- בעזרת מחלקות פנימיות ניתן להגדיר מחלקות אנונימיות – מחלקות ללא שם

- מחלקות אנונימיות שימושיות מאוד במערכות מונחות ארועים (כגון GUI) וילמדו בהמשך הקורס

הידור של מחלקות פנימיות

- המהדר (קומפיילר) יוצר קובץ .class עבור כל מחלקה. מחלקה פנימית אינה שונה במובן זה ממחלקה רגילה
- שם המחלקה הפנימית יהיה Outer$Inner.class
- אם המחלקה הפנימית אנונימית שם המחלקה שיוצר הקומפיילר יהיה Outer$1.class