





Adapter Design Pattern

מחשבים בחו"ל



■ יצאת לחו"ל (ברצלונה) עם המחשב הנייד



■ זכרת להביא כבל חשמל

■ תואם בהספק (220v)

■ תואם בתדר (50hz)



■ אבל כל זה לא מספיק,

■ אם לא הבאת מתאם



בעיית המתאם

■ הבעיה היא טכנית

- הלקוח (laptop) זקוק ל 220v ב- 50hz
- הספק (רשת החשמל) מספק 220v ב- 50hz
- הבעיה היא בממשק בין השניים

■ בעיה דומה עשויה לקרות גם בתוכנה

- במערכת חדשה אנו זקוקים ליכולות מסוימות (למחלקה שממשת משהו)
- קיים מודול (מחלקה) שנכתב למערכת קודמת המספק את היכולות חשובות
- אולם בשלב התכנון של המערכת החדשה הוגדר מנשק למחלקה זו השונה מהמנשק שאותו מממש המודול הקיים

דוגמא

```
public interface RandomNumberGenerator {  
    public int random();  
}
```

אפשר היה לממש את השירות בעצמנו, למשל כך:

```
public class FastRandom implements RandomNumberGenerator {  
  
    public int last;  
  
    public FastRandom() {  
        last = (int) System.nanoTime();  
    }  
  
    public int random() {  
        last = last * 1103515245 + 12345;  
        return (short) (last & 0xFFFF);  
    }  
  
}
```

שימוש חוזר

אולם הרבה יותר פשוט להשתמש במחלקה קיימת:

`java.util.Random` ■

בעיה: ■

המחלקה אינה מממשת את המנשק הדרוש במערכת: ■

`RandomNumberGenerator`

ובפרט היא איננה מממשת את השרות הדרוש `random()` ■

| Method Summary | |
|----------------|--|
| protected int | next (int bits) Generates the next pseudorandom number. |
| boolean | nextBoolean () Returns the next pseudorandom, uniformly distributed <code>boolean</code> value from this random number generator's sequence. |
| void | nextBytes (byte[] bytes) Generates random bytes and places them into a user-supplied byte array. |
| double | nextDouble () Returns the next pseudorandom, uniformly distributed <code>double</code> value between 0.0 and 1.0 from this random number generator's sequence. |
| float | nextFloat () Returns the next pseudorandom, uniformly distributed <code>float</code> value between 0.0 and 1.0 from this random number generator's sequence. |
| double | nextGaussian () Returns the next pseudorandom, Gaussian ("normally") distributed <code>double</code> value with mean 0.0 and standard deviation 1.0 from this random number generator's sequence. |
| int | nextInt () Returns the next pseudorandom, uniformly distributed <code>int</code> value from this random number generator's sequence. |
| int | nextInt (int n) Returns a pseudorandom, uniformly distributed <code>int</code> value between 0 (inclusive) and the specified value (exclusive), drawn from this random number generator's sequence. |
| long | nextLong () Returns the next pseudorandom, uniformly distributed <code>long</code> value from this random number generator's sequence. |
| void | setSeed (long seed) Sets the seed of this random number generator using a single <code>long</code> seed. |

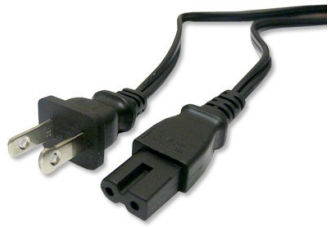
המתאם



- הבעיה היא טכנית והיא נעוצה בממשק:
 - המחלקה המבוקשת נדרשת לממש את `random()`
 - ואילו `java.util.Random`, אף על פי שהוא מממש אקראיות, הוא עושה זאת בעזרת פונקציות `nextXXX()`
- הפתרון: מחלקת מתאם (Adapter)

```
public class RandomAdapter implements RandomNumberGenerator {  
  
    java.util.Random r;  
  
    public RandomAdapter() {  
        r = new java.util.Random();  
    }  
  
    public int random() {  
        return r.nextInt();  
    }  
}
```

המחלקה מתאמת (מתרגמת) בין
המנשק של המחלקה `Random` למנשק
`RandomNumberGenerator`



לא רק תאומים

- ומה אם נרצה לצאת עם המחשב הנייד לארה"ב?
- כעת הבעיה היא לא רק בממשק
 - אלא גם בהפרשי המתחים
- ניתן לפתור בעיה זו באותה שיטה:
 - המתאם (adapter) יבצע גם את המרת הזרם (transformator) ע"י שימוש בספק המקורי
- בדוגמא שלנו: איך נממש מנשק חדש שבו נדרשים להחזיר זוג (Set) מספרים אקראיים?

```
import java.util.Set;

public interface RandomSetGenerator {
    public Set<Integer> random();
}
```

```
import java.util.Set;
import java.util.TreeSet;

public class RandomSetAdapter implements RandomSetGenerator {

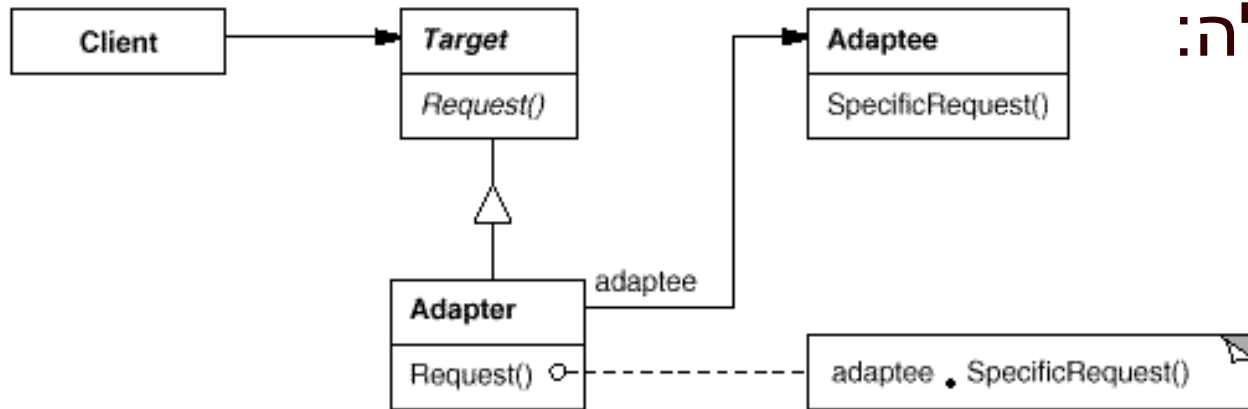
    java.util.Random r;

    public RandomSetAdapter() {
        r = new java.util.Random();
    }

    public Set<Integer> random() {
        Set<Integer> result = new TreeSet<Integer>();
        result.add(r.nextInt());
        result.add(r.nextInt());
        return result;
    }
}
```


סוגים של מתאמים

■ מבוססי הכללה:



■ מבוססי הורשה:

