Java lecturer 10

אז על מה נדבר היום...

- Final
- Enum
- תרגול מחלקות נוסף

RAVTECH

Final

עד היום ראינו שניתן לשנות כל משתנה במחלקה באמצעות גישה אליו ישירות או באמצעות מתודה פנימית שמשנה אותו.
לפעמים נרצה שמשתנה יהיה קבוע ולא ניתן לשינוי מסיבות שונות.

לפעמים זה משתנה שהמחלקה משתמשת בו לצרכים פנימיים בלבד ואין צורך לשנות אותו
או שזה משתנה שהמחלקה תספק אותו כי המשמעות שלו היא קבועה.

למשל משתנה מסוג int שמחזיר את מספר ימי השבוע.

אז במקרים כאלה נוסיף לפני שאנחנו מכריזים על סוג המשתנה את המילה השמורה **final**

במקרה כזה נצטרך בנוסף להכרזה גם לעשות השמה מיידית ולאתחל את ערך המשתנה.

לדוגמא:

```
public final int DAYS_OF_THE_WEEK = 7;
```

לאחר שביצענו למשתנה השמה, המשנה לא יהיה ניתן לשינוי בשום דרך.

*מוסכמה: משתנים קבועים נכתבים באותיות גדולות וקו תחתון מפריד בין המילים.

Final

ניתן דוגמא נוספת מתוך המחלקה Recipe שכתבנו.

יש לנו מתודה במחלקה שמקבלת את דירוג המתכון, ומגבילה את הדירוג בין 0 ל - 5
עד היום כתבנו אותה בצורה הבאה:

מכיוון שאנחנו יודעים שהמספרים הם קבועים נוכל לשנות אותם
ולהשתמש ב **final** בצורה הבאה:

```
private final int MAX_RATE = 5;
private final int MIN_RATE = 0;

public void setRate(int rate) {
    if (rate > MAX_RATE)
        this.rate = 5;
    else if (rate < MIN_RATE) {
        this.rate = 0;
    } else {
        this.rate = rate;
    }
}

public int getMaxRate() {
    return MAX_RATE;
}
```

RAVTECH

Final

מבחינה טכנית ההגדרה **final** אינה מוסיפה דבר ליכולות הביצועיות של השפה.
מה כן היתרונות שלה?

1. שימוש בקבועים יעזור לקוד להיות קריא יותר. אם במקום מספר או מילה מסויימת נציב קבוע ששמו הוא בעל משמעות יהיה קל הרבה יותר להבין את מטרת הקוד.
2. שימוש בקבועים עשוי לחסוך הרבה עבודה: אם משתמשים בנתון מסויים פעמים רבות אפשר לשנות את ערכו באמצעות שינוי אחד (של ערך הקבוע) במקום שינויים במקומות רבים בקוד.
3. שימוש בקבועים מקצר לפעמים את משך העבודה. במקום לכתוב את אותו משפט ארוך\מספר עשרוני מורכב פעמים רבות - אפשר להציב את הערך בקבוע ולהשתמש בו במקום.
4. הידיעה שהתוכן של משתנה או שדה לא ישתנה יותר עשויה מסייעת לקומפיילר לבצע את התוכנית בצורה יעילה יותר ובפחות זמן.

ניתן דוגמא לניסוי:
נכתוב 2 מתודות שמחברות בין 2 מחרוזות.

בשקף הבא ננסה להריץ אותם מיליון פעמים ברצף!
ונראה איזה מתודה צורכת יותר זמן בזמן ריצה.

```
public static String testFinal() {
    final String a = "a";
    final String b = "b";
    return a + b;
}
public static String testNonFinal() {
    String a = "a";
    String b = "b";
    return a + b;
}
```

Final

נסו בעצמכם

```
public static final int NUM_ITERATIONS = 1000000;

public static void main(String[] args) {

    long startTime, endTime;

    startTime = System.currentTimeMillis();
    for (int i = 0; i < NUM_ITERATIONS; i++)
        testFinal();
    endTime = System.currentTimeMillis() - startTime;
    System.out.println("Method with finals took " + endTime + " ms");

    startTime = System.currentTimeMillis();
    for (int i = 0; i < NUM_ITERATIONS; i++)
        testNonFinal();
    endTime = System.currentTimeMillis() - startTime;
    System.out.println("Method without finals took " + endTime + " ms");

}
```

RAVTECH

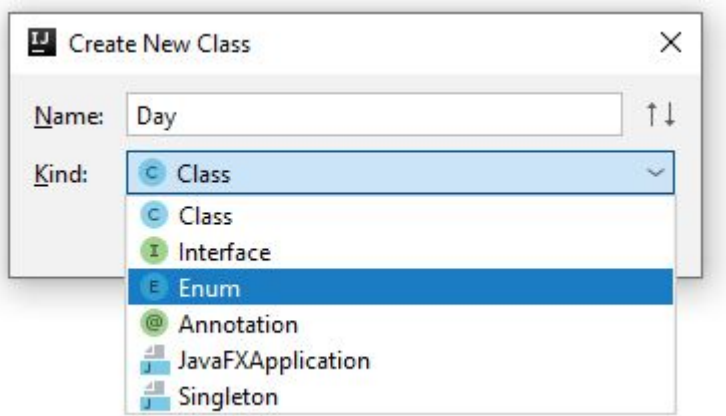
Enum

עד עכשיו ראינו שניתן ליצור משתנה קבוע בעל השמה אחת בלבד.
לפעמים גם נרצה ליצור משתנה קבוע עם קבוצה של השמות קבועים ומוגדרים מראש, והמשתנה יהיה חייב להיות מושם באחד מהערכים הללו בלבד
למשל אם נרצה ליצור משתנה שמייצג כיוון.
אז במקרה הזה אנחנו יודעים מראש שישנם רק ארבעה כיוונים. (צפון, דרום, מזרח, מערב)
או למשל שנרצה משתנה שמייצג רק את ימי השבוע.

עבור המקרה הזה יש סוג של מחלקה / משתנה שנקרא **enum**.
הדרך לשימוש בו היא פשוטה.

לצורך הדוגמא נרצה ליצור משתנה שמייצג את כשרות המתכון:

כמו שאנחנו יוצרים מחלקה חדשה אנחנו יכולים ליצור גם את ה **enum**.
ניצור מחלקה חדשה שנקרא לה Day
אבל בבחירת סוג המחלקה נבחר ב Enum



Enum

הקוד שלנו יראה כך:

```
public enum Day {
}
```

שלב שני נמלא את מחלקת ה enum בשמות ימות השבוע.

```
public enum Day {
    SUNDAY, MONDAY, TUESDAY, WEDNESDAY, THURSDAY, FRIDAY, SATURDAY
}
```

בעצם יצרנו כאן קבוצה של משתנים קבועים שאין להם ערך אבל כל אחד ואחד מהם מייצג יום בשבוע
אבל מי שירצה להשתמש באובייקט מסוג Day יהיה חייב לעשות השמה לאחד מהמשתנים האלו.
וזה יראה לדוגמא כך:

```
Day day = Day.FRIDAY;
```

נראה עוד דוגמאות לשימוש:

Enum

אם יש לנו פונקציה שמקבלת בתור פרמטר אובייקט מסוג Day
מי שיקרא לה יהיה חייב לספק את אחד מהמשתנים של Day.

הבעיה היא שהפונקציה קצת משעממת....
אז נוסיף לה יכולות

```
public static void getDay(Day day) {

    switch (day) {
        case SUNDAY:
            System.out.println("Java");
            break;
        case MONDAY:
            System.out.println("Exercises");
            break;
        case TUESDAY:
            System.out.println("Java");
            break;
        case WEDNESDAY:
            System.out.println("Exercises");
            break;
        case THURSDAY:
            System.out.println("Java");
            break;
        ....
    }
}
```

RAVTECH

תרגול

בתרגול הבא נבנה תוכנה לניהול בית ספר.

למחלקה Scool יהיו את היכולות הבאים:

- תכונות:
  - שם
  - מנהל
  - כתובת
  - מספר טלפון
  - רשימת חוגים
  - רשימת כיתות
- בנאים:
  - בנאי ברירת מחדל בלבד
- פעולות:
  - Getters and Setters
  - הוספת כיתה
  - הוספת חוגים
  - הרשמת תלמיד לכיתה
  - הרשמת מורה לכיתה
  - קבלת רשימת חוגים
  - קבלת פרטי חוג לפי שם
- המשך פעולות:
  - קבלת מספר התלמידים הכללי
  - קבלת מספר התלמידים לפי כיתה / שם חוג
  - קבלת רשימת התלמידים שעדיין לא שילמו
  - קבלת רשימת תלמידים לפי שם משפחה
  - קבלת פרטי תלמיד לפי מספר זהות
  - קבלת רשימת מורים
  - קבלת פרטי מורה לפי שם פרטי ושם משפחה
  - קבלת ממוצע ציונים של כל התלמידים בכיתה מסוימת
  - קבלת ממוצע ציונים של כל התלמידים בבית הספר
  - קבלת רשימת הכיתות שמספר התלמידים חורג מהכמות המקסימלית

RAVTECH

תרגול

בניית מחלקה לכיתה.

למחלקה Grade יהיו את היכולות הבאים:

- תכונות:
  - רמת כיתה (לפי הטבלה המצורפת)
  - רשימת תלמידים
  - כמות מקסימלית של תלמידים
  - מורה
- בנאים:
  - בנאי ברירת מחדל בלבד
- פעולות:
  - Getters and Setters
  - הוספת תלמיד (רק אם הוא תואם לרמה)
  - הסרת תלמיד לפי מספר זהות
  - קבלת ממוצע ציונים כיתתי
  - הוספת מורה

גיל	כיתה
6–7	1 / א
7–8	2 / ב
8–9	3 / ג
9–10	4 / ד
10–11	5 / ה
11–12	6 / ו
12–13	7 / ז
13–14	8 / ח
14–15	9 / ט
15–16	10 / י
16–17	11 / יא
17–18	12 / יב

תרגול

בניית מחלקה לתלמיד / מורה

למחלקה Person יהיו את היכולות הבאים:

- תכונות:
  - מורה, תלמיד או מנהל
  - שם פרטי
  - שם משפחה
  - מספר זהות
  - גיל
  - כתובת
  - מספר טלפון
  - האם שילם (מורה ומנהל נחשבים ששילמו)
  - ממוצע ציונים
- בנאים:
  - בנאי ברירת מחדל בלבד
- פעולות:
  - Getters and Setters

RAVTECH

תרגול

בניית מחלקה לחוג

למחלקה Course יהיו את היכולות הבאים:

- תכונות:
  - שם מקצוע
  - שעת התחלה
  - שעת סיום
  - מחיר
  - רשימת תלמידים
  - כמות מקסימלית של תלמידים
  - מורה
- בנאים:
  - בנאי ברירת מחדל בלבד
- פעולות:
  - Getters and Setters
  - הוספת תלמיד (אין אפשרות להוסיף את אותו התלמיד פעמיים)
  - הסרת תלמיד לפי מספר זהות