תאריך: 19.2.2026

מבוא למדעי המחשב – כתה יא-1

תכנות מונחה עצמים – ירושה Inheritance + מפסיקים לשכפל קוד

תירגול #2

מבוא:

בשיעור הקודם ראינו שכאשר הוספנו את המחלקה `Cat`, נאלצנו להעתיק כמעט את כל הקוד ממחלקת `Dog`.

מצב כזה נקרא כפילות קוד, והוא מצב לא רצוי.

כאשר משנים משהו קטן (למשל איך אנחנו מייצגים את השם – שם הכלב או שם + סוג), צריך לזכור לעדכן אותו בכמה מחלקות שונות.

ככל שמספר המחלקות גדל – כך גדל גם הסיכון לטעויות. וגם אם לא נעשה טעויות – זה ממש מעצבן לעבור על עשרות קבצים ולעשות את אותו השינוי.

בשיעור זה נלמד עיקרון חשוב בתכנות מונחה עצמים הנקרא ירושה (`Inheritance`).

הרעיון פשוט: במקום לשכפל קוד, נוציא את כל מה שמשותף למחלקות השונות אל מחלקה אחת משותפת.

בדומה למתמטיקה, כאשר מוציאים גורם משותף מחוץ לסוגריים – גם כאן נרכז את המשותף במקום אחד, ואת החלקים הייחודיים נשאיר בכל מחלקה בנפרד.

מטרות

1. להבין מדוע כפילות קוד היא בעיה.
2. ללבנות מחלקת בסיס (base) שמרכזת קוד משותף.
3. לכתוב מחלקות יורשות המשתמשות ב-base בבנאי.
4. לראות כיצד שינוי במקום אחד משפיע על כמה מחלקות.

<u>תחנה 1 – "זיהוי המשותף"</u>

נעיין במחלקות `Cat, Dog` מהשיעור הקודם – עיינו בקובץ `Pet_Example_4.cs`.

1) כתבו בטבלה מה משותף ומה שונה בין המחלקות (התייחסו למשתנים ופעולות):

משותף	שונה בין המחלקות

רמזים: שדות (שם וגיל), פעולות (`GetName, GetAge, HaveBirthday, ToInfo`)

2) כתבו במילים שלכם (2–3 שורות): מה הבעיה בכך שיש קוד משותף שמופיע פעמיים?

3) שאלה לחשיבה: אם מחר נוסיף Parrot ועוד Rabbit – כמה פעמים נצטרך להעתיק? וכמה מקומות נצטרך לשנות כאשר מבצעים שינוי קטן?

<u>הרעיון המרכזי</u>

את מה שמשותף – משתנים ופעולות, נרכז במחלקה אחת משותפת, שנכנה אותה מחלקת בסיס `Base Class`. (זה כינוי. למחלקה בקוד נתן שם משמעותי). המחלקות האחרות יקראו: "מחלקות יורשות"

במקרה שלנו, כפי שנראה בתחנה הבאה, ניצור מחלקת בסיס שנקראת `Class Pet`.

תחנה 2 – בניית מחלקת בסיס Pet

המטרה: להעביר את הקוד המשותף ממחלקות Dog ו־Cat אל מחלקה אחת משותפת.

משימה:

1) צרו מחלקה חדשה בשם Pet.
2) העבירו אליה את כל השדות המשותפים:
   - שם החיה
   - גיל (בחודשים)
   - סוג החיה (אם קיים)
3) כתבו פעולה בונה למחלקה Pet בצורה הבאה:

```
public Pet(string name, int ageInYears, string species)
```

בתוך הפעולה הבונה: (א) שמרו את השם, (ב) שמרו את הסוג, (ג) המירו את הגיל לשמירה בחודשים

4) העבירו אל המחלקה Pet את הפעולות המשותפות:
   - GetName()
   - GetAge()
   - HaveBirthday()
   - ToInfo()

בשלב זה – מחלקות Dog ו־Cat עדיין קיימות כפי שהן.

כיצד נגרום ל Cat, Dog להשתמש בקוד של Pet?

תחנה 3 – ירושה בפועל – המחלקות Dog ו־Cat יורשות מ־Pet

המטרה: לגרום למחלקות של Dog ו- Cat להשתמש בקוד המשותף שנמצא במחלקה Pet, במקום לשכפל אותו.

משימה:

1) עדכנו את כותרת המחלקה Dog ו- Cat כך שתירש מ־ Pet:

```
class Dog : Pet
class Cat : Pet
```

2) כעת, מכיוון שהשדות והפעולות המשותפים נמצאים כבר ב־Pet, מחקו מתוך Dog ומתוך Cat את כל מה שעבר ל־Pet:

- שדות משותפים (שם, גיל, סוג)
- פעולות משותפות(GetName, GetAge, HaveBirthday, ToInfo)

בשלב הזה, בכל מחלקה יורשת אמורה להישאר **רק פעולה בונה.**

פעולה בונה במחלקה יורשת – חובה להשתמש ב־base

3) ב־ Dog ו- Cat כתבו פעולה בונה כך:

```
public Dog(string name, int ageInYears) : base(name, ageInYears, "Dog")
{
}
public Cat(string name, int ageInYears) : base(name, ageInYears, "Cat")
{
}
```

הסבר קצר:
הכתיבה base(...) מפעילה את **הפעולה הבונה** של מחלקת הבסיס (Pet) ומעבירה אליה את הנתונים הדרושים.

בדיקה

4) עברו ל־ Main_Dog_And_Cat() או ה־Main של הפרוייקט. הריצו וודאו שהתכנית עובדת ומתקבל הפלט הרצוי.

<u>תחנה 4 – בדיקה אצל הווטרינר</u>

נוסיף למחלקת Pet את הפעולה הבאה:

```
public void Talk(string sound)
{
    Console.WriteLine(GetName() + " says: " + sound);
}
```

כעת נוסיף למחלקת Vet פעולה חדשה:

```
public void CheckPet(Pet p)
{
    // הווטרינר "בודק" את החיה בכך שהוא מבקש ממנה לדבר
}
```

וב־Main נקרא לדוגמה:

```
Vet v = new Vet();
v.CheckPet(d);
v.CheckPet(c);
```

משימה:

1) ממשו את הפעולה CheckPet כך שהווטרינר יגרום לחיה לדבר. לדוגמה: כלב צריך לומר: "Warf Warf". חתול צריך לומר: "Meow"
2) ב. שימו לב: אם הווטרינר צריך לבדוק גם כלב וגם חתול — כמה פעולות CheckPet תצטרכו להוסיף במחלקת Vet?
3) ממשו את הקוד כפי שאתם חושבים שנכון.

<u>שאלה למחשבה: (לא לפתור אותן בקוד כרגע)</u>

האם הקוד של הווטרינר צריך לדעת איזה צליל משמיעה כל חיה? או שהקוד של כל חייה צריכה להיות אחראית להשמיע את הקול הנכון?

האם ייתכן שנכתוב בטעות בקוד של הווטרינר צליל לא מתאים?

האם יש דרך לגרום לכל חיה לדעת בעצמה איך לדבר?

<u>תרגיל ירושה – צורות המחלקה Shape</u>

התוכנית מדגימה שימוש בירושה באמצעות מחלקת בסיס בשם Shape המייצגת צורה כללית בעלת צבע, וממנה יורשות המחלקות Rectangle ו־Circle. מחלקת הבסיס מכילה את הנתונים והפעולות המשותפים לכל הצורות (כגון צבע), ואילו המחלקות היורשות מוסיפות שדות ופעולות ייחודיים לכל סוג צורה (מידות וחישוב שטח). בתוכנית הראשית נוצרים אובייקטים מסוג מלבן ועיגול, ומודפס עבור כל אחד מהם מידע הכולל את הצבע ואת השטח המחושב, ובכך מודגם כיצד קוד משותף מרוכז במחלקת בסיס אחת ומשרת מספר מחלקות שונות.

<u>הוראות</u>

צרו פרוייקט חדש ב Visual Studio.

הורידו את הקבצים מה GDRIVE <u>בקישור הזה</u>:

https://drive.google.com/drive/folders/1Jd6Zim-Dp7uLvY_fXXLSX2FdqZlxkTM7?usp=drive_link

עליכם לממש את המחלקות השונות Shape, Circle, Rectangle על פי ההערות בקבצים.

במקביל הוציאו מהערה את השורות המתאימות ב Main והריצו את התכנית.

<u>הערה לגבי סדר העבודה:</u>

1) כתבו את המחלקה SHAPE ואז תריצו את התכנית (לא לשכוח להוציא את השורות המתאימות מהערה ב MAIN). הריצו וודאו שזה עובד היטב.
2) לאחר מכן Rectangle.
3) לאחר מכן Circle.