תרגיל מס' 4 – ממשקים, משלחות, אירועים ותפריטים

מטרות

- תכנות מונחה עצמים ו- polymorphism
- הטמעה של עבודה עם ממשקים - interfaces
- הטמעה של עבודה עם מצביעים לפונקציות ומשלחות (delegates) – Action<T>

ידע נדרש

- עבודה עם ממשקים - interfaces
- עבודה עם משלחות – delegates (Action<T>)
- שימוש ב- Collections
- עבודה עם מספר פרויקטים

התרגיל

עליכם לממש רכיב תוכנה שמסייע בהצגה וניהול של תפריטים היררכיים לאפליקציות Console.
(אם היה כזה רכיב, הייתם יכולים למשל להשתמש בו בתרגיל מס' 3 כדי להציג את התפריטים של המערכת למשתמש, במקום לכתוב בעצמכם את הקוד שמציג אותם ומנהל את בחירות המשתמש כפי שבטח עשיתם בתרגילים רבים בעבר..)

עליכם לממש מחלקה בשם MainMenu אשר תאפשר (לתוכניתן שישתמש בה) לבנות תפריט לפי צרכיו (למשל במערכת לניהול מוסך שמימשתם בתרגיל הקודם), ע"י הגדרה של פריטי-תפריט בתפריט הראשי או לכל תת-תפריט.

אפליקציה שרוצה להציג תפריטים למשתמש (למשל האפליקציה לניהול מוסך מהתרגיל הקודם) צריכה להחזיק מופע של מחלקת MainMenu.

מחלקת MainMenu תאפשר למשתמש בה (למשל התוכניתן שבונה את האפליקציה של המוסך) **לבנות את היררכיית התפריטים** (נניח בשלב האתחול באפליקציה שלו) ע"י הגדרת והוספת <u>פריטים</u> (מופעים של מחלקה בשם **MenuItem**) לתפריט ותת-פריטים לכל פריט וכן הלאה..

כדוגמא לתפריט עם פריטי תפריט שבנוי בצורה היררכית, קחו למשל את התפריט של visual studio – התפריט הראשי נמצא בראש החלון ומכיל מספר פריטים ברמה הראשונה (File, Edit, View..) שמכילים פריטים בעצמם. לחיצה על חלק מהפריטים גורמת לפעולה מסוימת בתוכנה. לחיצה על פריטים אחרים גורמת להופעה של תת-תפריט (פריטי תפריט נוספים שמאוגדים מתחת לפריט שבחרנו) וכן הלאה..

המשך בעמוד הבא..

אפיון

כאמור, עליכם לממש רכיב תוכנה שמספק הצגה וניהול של תפריטים היררכיים לאפליקציות Console.

הפעלת הפונקציה m_MainMenu.Show() תגרום להצגת התפריט הראשי (הרמה הראשונה) ובפועל תקיים את הלולאה שתהייה בפועל הלולאה הראשית שכל אפליקציית console זקוקה לה כדי להתקיים.

בכל שלב:

1. תוצג הרמה הנוכחית של התפריט למשתמש
2. תוצג בקשה למשתמש לבחור באחד הפריטים.
3. תקלט הבחירה מהמשתמש
4. תתבצע בדיקת תקינות קלט + הצגת הודעה מתאימה במקרה של קלט לא תקין
5. ניווט לתת-תפריט / הפעלת הפעולה שנבחרה:
   a. בחירה בפריט שמכיל פריטי משנה תנקה את המסך ותציג את פריטי המשנה הרלבנטיים (כלומר את הרמה הבאה בתפריט, כלומר חזרה לסעיף 1).
   b. בחירה בפריט שאינו מכיל פריטי משנה, כלומר זו בחירה שבגינה צריכה לקרות פעולה כלשהיא במערכת (למשל –האפשרות להכניס רכב חדש למוסך) תנקה את המסך ותפעיל את הפונקציה במערכת שמספקת את ממשק המשתמש שתומך בפונקציונאליות הזאת של המערכת. לאחר השלמת הפעולה במערכת, תוצג שוב הרמה של התפריט שבה היינו קודם וחוזר חלילה עד לרגע שהמשתמש ייבחר באופציית יציאה מהתוכנית (ע"י פריט מיוחד בתפריט)

<u>שימו לב: התפריט מציג באופן קבוע בסדר הזה (בכל רמה שהיא):</u>

- כותרת (ברמה הראשונה - כותרת ראשית, בכל רמה אחרת – כותרת הפריט שנבחר ברמה הקודמת)
- רשימת פריטים ששייכים לרמה הנוכחית (ממוספרים מ-1)
- פריט שכתוב בו 'Back' (או 'Exit' במידה וזה התפריט הראשי) (ממוספר 0)
- הודעה שמבקשת מהמשתמש לבחור מבין הפריטים או לבחור בפריט 'Back' / 'Exit'

```
** Drinks Machine **
--------------------
1. Tea
2. Coffee
3. Juice
0. Exit
Please enter your choice (1-3 or 0 to exit):
>> 2
```

<Console.Clear()>

```
** Coffee **
------------
1. Americano
2. Espresso
3. Capuchino
4. Late
0. Back
Please enter your choice (1-4 or 0 to go back):
>> 3
```

המערכת שמשתמשת במחלקה MainMenu תציג כעת למשתמש את ממשק המשתמש הרלוונטי להכנת אספרסו, ובסיום הפעולה תוצג שוב הרמה הנוכחית בתפריט:

הסברים נוספים:

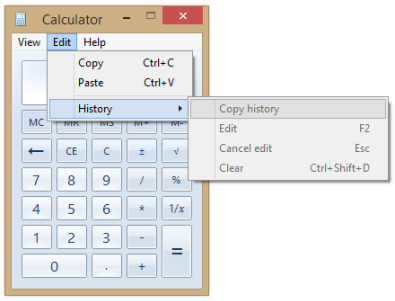
הרכיב מאפשר למי שמשתמש בו (מתכנת אחר, שבונה למשל מערכת לניהול מוסך לקונסול...) לבנות תפריט לפי צרכיו, עם תתי תפריטים בהתאם, ולהשתמש בו במקום לממש את התפריט למשתמש בעצמו.

הדרך לעשות את זה היא ע"י כך שהרכיב יחזיק אוסף של פריטי תפריט, וכל פריט יחזיק אוסף של תתי-פריטים במקרה הצורך (שיהוו את תת-התפריט שלו) וכן הלאה.

במערכת שמשתמשת במנגנון התפריטים הזה (למשל המערכת לניהול מוסך), צריכה להיות פונקציה שמאתחלת את מבנה התפריט שלה.

לגבי הפונקציות שעל הפריטים הסופיים (ה"עלים") להפעיל - צריכה אכן להיות דרך לאפשר להם להפעיל את הפונקציה המתאימה באפליקציה (פריט סופי = פריט שאין לו תתי-פריטים, כלומר פריט שמטרתו להפעיל פעולה מסוימת ולא להציג תת-תפריט).

קחו לדוגמא את התפריט המוצג במחשבון (במקרה הזה: 3 פריטים בתפריט, 3 פריטים בפריט השני, 4 פריטים בפריט השלישי שלו..):



הוראות למימוש

עליכם לממש זאת בשתי הטכניקות שהוסברו לפתרון בעיית "עובד מודיע שהוא חולה"
(רמז – במקרה הזה, פריט בתפריט מודיע למערכת שבחרו בו)

1. ע"י שימוש בממשקים – interface
איזה ממשק יש להגדיר? אילו פעולה/ות הוא יחשוף? היכן (באיזה אסמבלי) יש להגדיר את הממשק? מי אמור לממש את הממשק? מי אמור להשתמש בממשק? מה צריכה להיות רמת החשיפה שלו (access modifier)? כיצד תקבל המחלקה שמשתמשת בממשק את האובייקט שמממש את הממשק? וכו'.

2. ע"י שימוש ב- Action<T>
מי יחזיק מופע שלו? מי יפעיל אותו? מי מייצר מופע שלו? וכו'.

פתרון התרגיל יכיל solution ובו 3 פרויקטים:

1. **Ex04.Menus. Interfaces** - יכיל מימוש לפי הטכניקה הראשונה.
זהו **אינו** פרוייקט שמייצר קובץ הרצה אלא ספריית מחלקות בלבד (.NET Assembly)

2. **Ex04.Menus.Events** - יכיל מימוש לפי הטכניקה השנייה.
זהו **אינו** פרוייקט שמייצר קובץ הרצה אלא ספריית מחלקות בלבד (.NET Assembly)

3. **Ex04.Menus.Test** – אפליקציה (EXE) שמדגימה שימוש בכל אחד משני הפרויקטים מהסעיפים הקודמים:

במתודה Main יש לבנות **שני** תפריטים בעלי 2 רמות כל אחד (אחד עבור כל אחת משתי השיטות).
תחילה יוצג התפריט הראשון והמשתמש יוכל לנווט בו. כשהמשתמש יבחר לצאת מהתפריט (Exit) יוצג לו התפריט השני.

להלן המפרט של התפריטים:

ברמה הראשונה יהיו 2 פריטים.

a. פריט 1 יציג תת-תפריט עם הכותרת – "Letters and Version" שבו יהיו 2 פריטים:

   i. פריט פעולה – "Show Version"
   מפעיל (בעקיפין)מתודה ב- Test שמציגה את הטקסט הבא:
   App Version: 25.2.4.4480

   ii. פריט פעולה – "Count Lowercase Letters"
   מפעיל (בעקיפין) מתודה ב- Test שמבקשת מהמשתמש לכתוב משפט והמערכת אומרת לו כמה אותיות קטנות (lower-case) יש במשפט.

b. פריט 2 יציג תת-תפריט עם הכותרת "Show Current Date/Time" שבו יהיו 2 פריטים:

   i. הפריט השני יהיה פריט פעולה – "Show Current Date"
   מפעיל (בעקיפין) מתודה ב- Test שמציגה את התאריך של היום.

   ii. הפריט הראשון יהיה פריט פעולה – "Show Current Time"
   מפעיל (בעקיפין) מתודה ב- Test שמציגה את השעה נוכחית.

שימו לב, **אין צורך להימנע משיכפול קוד** בין שני הפרויקטים הראשונים.
נא להתייחס אליהם כאל שני תרגילים שונים, העשויים להכיל קטעי קוד דומים.

פלט לדוגמא:

```
** Delegates Main Menu **
------------------------
1. Letters and Version
2. Show Current Date/Time
0. Exit
Please enter your choice (1-2 or 0 to exit):
>> 1

<clear screen>

** Letters and Version **
--------------------------
1. Show Version
2. Count Lowercase Letters
0. Back
Please enter your choice (1-2 or 0 to go back):
>> 1
App Version: 25.2.4.4480

** Letters and Version **
--------------------------
1. Show Version
2. Count Lowercase Letters
0. Back
Please enter your choice (1-2 or 0 to go back):
>> 2
How are you?
> There are 8 lowercase letters in you text

** Letters and Version **
--------------------------
1. Show Version
2. Count Lowercase Letters
0. Back
Please enter your choice (1-2 or 0 to go back):
>> 0

<clear screen>

** Delegates Main Menu **
------------------------
1. Letters and Version
2. Show Current Date/Time
0. Exit
Please enter your choice (1-2 or 0 to exit):
>> 2

** Show Current Date/Time **
------------------------
1. Show Current Date
2. Show Current Time
0. Back
Please enter your choice (1-2 or 0 to go back):
>> 2
> Current Time is 15:35
```

הערות נוספות

1. יש לתכנן את המערכת בצורה הטובה ביותר של קוד מבחינת חלוקה למחלקות ולמתודות, הורשה ופולימורפיזם. יש להימנע משכפול קוד/לוגיקה ע"י חלוקה למתודות והפרדה של לוגיקה לרכיבים. שימו לב, אלו הם רכיבים שמממשים מערכת תפריטים **לקונסול.** לכן מן הסתם מותר להם לכתוב ולקרוא מהקונסול.
2. ניתן להיעזר בקבוצת הפייסבוק של הקורס כדי לשאול שאלות בנוגע לתרגיל.
3. יש לעמוד בתקנים לכתיבת קוד כפי שמפורט במסמך הרלוונטי שניתן למצא באתר הקורס. נקודות ירדו למי שלא יעמוד בתקנים אלו.
4. **שימו לב! המסמך "Coding Standards" מכיל סעיף בנוגע למוסכמות בכתיבת קוד שקשור למשלחות ואירועים (delegates ו- events). נא לקרוא ולעמוד במוסכמות אלו.**
5. יש לעמוד בהוראות ההגשה כפי שמפורט במסמך הרלוונטי שניתן למצא באתר הקורס. נקודות ירדו למי שלא יפעל לפי הוראות אלה. אם יש אי הבנה ניתן לשאול בקבוצת הפייסבוק.
6. נא להימנע מהעתקות (הן מתגלות מאוד בקלות).

בהצלחה!