כתיבת מחלקה (Class)

ציוד נדרש:

- ערכת פיתוח

רקע עיוני

- הבנה מעמיקה בנישא תכנות מונחה עצמים (OOP)
- מרכיבים בסיסיים של אובייקטים וסוגיהם:
  - constructor
  - שדה (field)
  - מאפיינים (Properties)
  - מטודות / פונקציות (Method / function)
- ירושה ו overwrite (אופציונאלי)

מבוא

עד כה דיברנו על משתנים מסוגים שונים כמו:

- String
- Int
- double
- long
- bool
- ועוד

כאשר נרצה להשתמש בטיפוסים מורכבים יותר מאשר משתנה פשוט, כדאי להשתמש בשפת התכנות באופן מיטבי. שפת C שקדמה ל # C היינה בעלת טיפוסים מסוג מבנה (Structure) שיכול להכיל מספר משתנים מסוגים שונים בוזמנית. מחלקה (class) של C# היא למעשה הרחבה של התפיסה. המחלקה יכולה להכיל לא רק משתנים מסוגים שונים, אלא גם פונקציות / מטודות. את המחלקות ניתן ליצור בדרכים שונות תוך שימוש בפעולות בונות שונות שלהם. כאשר נבצע פעולת קריאה או כתיבת נתונים מתוך המחלקה, היא גם יכולה לבצע אוטומטית פעולות שנגדיר על נתונים אלו או אחרים. למשל, בכתיבת נתונים לתוך מאפיין שנת לידה, לדוגמא, נרצה לוודא שהערך המוזן אליו היינו חוקי והגיוני שהרי לא תיתכן שנת לידה שגדולה מהשנה הנוכחית. אפשרויות אלו של מחלקות משנות את כל התפיסה של כתיבת התוכנה והופכות אותה מפרוצדוראלית (מבוססת על פונקציות) למונחת עצמים (מבוססת על מחלקות ועצמים / אובייקטים (objects)). מכאן גם ששפת # C הינה שפה תכנות המאפשרת תכנות מונחה עצמים (OOP - Object Oriented Programming).

מרכיבי מחלקה עיקריים הם:

- שדות – משתנים רגילים כמו אלו שאנו כבר מכירים.
- מאפיינים – "שער גישה" לשדה מסוים במטרה לקרוא את המידע שבתוכו או לכתוב אותו.
- פעולות בונות – או באנגלית constructors (קונסטראקטורים) היוצרים את ההעתק של המחלקה ע"פ בקשתנו. פעולה זו דומה להגדרת משתנה, אלא שהפעלת הקונסטראקטור לא מגדירה, אלא יוצרת את המחלקה שהוגדרה.
- פונקציות, או כפי שמכנים אותם אותם ב C# מטודות (methods) – אלו הן בדומה לפונקציות של שפות אחרות, אוסף של פעולות המתבצעות בזו אחר זו ומשמשות למטרה מוגדרת.

בכדי להמחיש עניין המחלקות נבנה מחלקה המשמיעה תווים.

מהלך הניסוי

1. נפתח פרויקט חדש של C# עבור ערכת NOA.
2. בתוך קובץ program.cs תתקבל התוכנה הבאה:

```
using System;
using Microsoft.SPOT;
using Microsoft.SPOT.Hardware;
using Stm32;
using System.Threading;
using System.Text;

namespace NOA_Application
{
    public class Program
    {
        public static void Main()
        {
            Debug.Print("Hello World! This is NOA.");
            Thread.Sleep(Timeout.Infinite);
        }
    }
}
```

3. נשים לב לכך שתוכנה שלנו בעצם כבר כוללת מחלקה (class) אחת. השם שלה הוא Program ובתוכה נמצאת הפונקציה הראשית Main().
4. המחלקה Program היינה ציבורית (public) ולכן ניתן לגשת אליה גם מחוץ ל namespace בו היא מוגדרת.

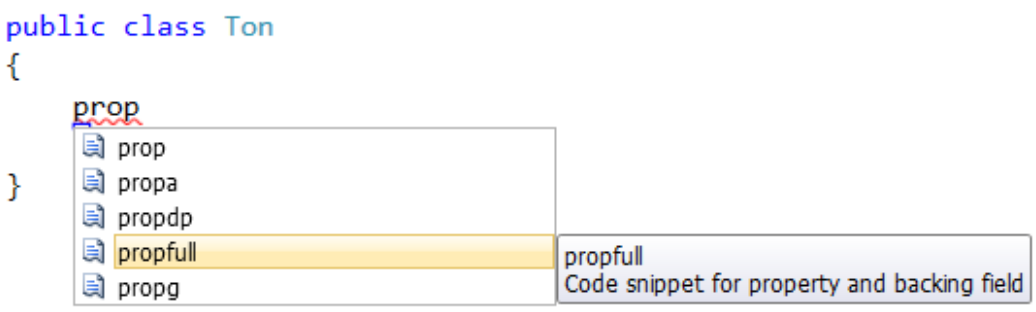
5. נגדיר מחלקה חדשה שלנו וניתן לה את השם Ton על דרך שהוגדרה המחלקה הקיימת:

```
using System;
using Microsoft.SPOT;
using Microsoft.SPOT.Hardware;
using Stm32;
using System.Threading;
using System.Text;

namespace NOA_Application
{
    public class Ton
    {
    }

    public class Program
    {
        public static void Main()
        {
            Debug.Print("Hello World! This is NOA.");
            Thread.Sleep(Timeout.Infinite);
        }
    }
}
```

6. המחלקה שהוגדרה היינה ריקה כרגע ועלינו למלא אותה בתוכן הרצוי. ניתן לעשות זאת באופן ידני, אך קל יותר להשתמש ב snippet code (קטע קוד שנכתב מראש למטרה מסוימת וניתן "לשתול" אותו בדומה לפעולת "הדבק" בעורך תמלילים). לשם כך:
   א. בתוך המחלקה, נקליד את הטקסט prop (קיצור של property - מאפיין)
   ב. מתוך הרשימה הנפתחת נבחר עם עכבר או עם החצים של המקלדת + enter את האופציה של propfull.



ג. המילה propfull תופיע על המסך.

ד. נקיש על המקש Tab שבצידה השמאלי של המקלדת. המילה propfull תוחלף אוטומטית בקוד הבא:

```
1.  private int myVar;
2.  public int MyProperty
3.  {
4.      get { return myVar; }
5.      set { myVar = value; }
6.  }
```

בשורה מס' 1 מופיעה הגדרת משתנה (שדה/ field) מסוג int בשם myVar שזהו שם ברירת מחדל. משתנה זה מוגדר כמשתנה פרטי (private), כלומר לא ניתן לגשת אליו מחוץ למחלקה בה הוא מוגדר.

בשורה מס' 2 מתחילה הגדרה של מאפיין (property) המחלקה. המאפיין צריך להיות ציבורי (public) בכדי שיהיה ניתן לגשת אליו (לקרוא ממנו או לכתוב אליו) מחוץ למחלקה בה הוא מוגדר. גם המאפיין הוא מסוג int והשם ברירת מחדל שניתן לו הוא MyProperty.

בשורה מס' 4 מופיעות הגדרות של דרכי הקריאה של המאפיין ובשורה מס' 5 מוגדרות דרכי ראיה מהמאפיין. לא נשנה הגדרות אלו בשלב זה של הלימוד, אך ניתן לשים לב לכך שבשורה 4 המאפיין מחזיר את הערך ששמור בתוך המשתנה (שדה) myVar שהוגדר יחד עם המאפיין ובשורה 5 הערך אותו מכניסים למאפיין נשמר בתוך אותו המשתנה myVar.

7. נתאים את הקטע קוד הנ"ל לצרכים שלנו:

א. נשנה את הסוג של <u>המשתנה</u> ושל <u>המאפיין</u> מ int ל string.

ב. נשנה את השם של המשתנה מ myVar ל note (משמעותה באנגלית היא גם "תו מוזיקאלי" ולא רק "הערה"). יש לעשות את השינוי בכל המקומות בהם המשתנה מוזכר.

ג. נשנה את השם של המאפיין משם ברירת מחדל שלו, MyProperty, ל Note.

שימו לב: C# היא שפה key sensitive, כלומר מבדיל בין אותיות גדולות וקטנות, לכן חשוב להקפיד לרשום את האותיות הגדולות כגדולות וקטנות כקטנות.

למרות שישנם הרגלים שונים בעניין, אנו ככלל נאמץ לעצמנו גישה לפיה למשתנים אנו נותנים שמות המתחילים באותיות קטנות ולמאפיינים שמות המתחילים באותיות גדולות.

8. בסה"כ יתקבל הקוד הבא של המחלקה אותה כתבנו:

```
public class Ton
{
    private string note;

    public string Note
    {
        get { return note; }
        set { note = value; }
    }

}
```

9. המחלקה Ton מכילה כרגע מאפיין אחד בשם note ושדה אחד בשם Note.
10. נוסיף למחלקה את הפעולה הבונה / הבנאי (constructor) שבעזרתו נוכל ליצור עותקים של המחלקה שאת הגדרתה אנו כותבים כאן. הבנאי יגדיר את הערכים של כל השדות של המחלקה.
    א. בסוף, לפני סגירת הסוגריים המסולסלות של המחלקה נוסיף את הקוד הבא:

```
1.  public Ton()
2.  {
3.      note = "do";
4.  }
```

בשורה הראשונה אנו מגדירים את הפעולה הבונה כציבורית בכדי שנוכל להשתמש בה מחוץ למחלקה. הסוגריים העגולות ריקות, כלומר הבנאי לא יקבל מהמפעיל שו מידע ועליו לקבוע את ערכי ברירת המחדל לכל השדות שבתוכו. לשם כך, בשורה השלישית, אנו מכניסים את הטקסט do לתוך השדה note. זה יהיה תו ברירת מחדל עד אשר יחליטו לשנות אותו.

ב. נוסיף בנאי נוסף שבשונה מהקודם, כן מקבל ערך של התו הרצוי מהמפעיל שלו:

```
1.  public Ton(string Note)
2.  {
3.      note = Note;
4.  }
```

בנאי זה יוקבל מהמפעיל שלו את הערך של התו אותו מייצרים וישמור אותו בתוך השדה note.

שימו לב: אין קשר בין המאפיין Note של המחלקה לבין הפרמטר Note אותו מקבל הבנאי.

11. נגדיר את הפונקציה (מטודה / פעולה) שתציג על המסך בחלונית ה output את התו. לשם כך, לאחר שני הבנאים, נוסיף את הקוד הבא:

```
public void Display()
{
    Debug.Print("The note is: " + note);
}
```

ניתן לראות שהפונקציה מציגה על המסך את הטקסט `The note is:` ואחריו בשרשור יוצג התוכן של השדה note.

.12

.13

.14

.15

.16

.17

אנחנו רוצים להגדיר אובייקט מהסוג של NewTon.

```
Ton NewTon = new Ton("DO");
```

כאשר NewTon הוא מסוג אוביקט של Ton

והוא מאתחל את הבנאי במילה DO

אחרי בניית האובייקט אנחנו רוצים להפעיל את ההתנהגות של המחלקה . למשל PrintTon

נקרא לאוביקט NewTon ונבקש להפעיל את המטודה/הפונקציה PrintTon();

```
NewTon.PrintTon();
```

התוכנה הכוללת נכתבת כך :

```
using System;
using Microsoft.SPOT;
using Microsoft.SPOT.Hardware;
using Stm32;
using System.Threading;
using System.Text;

namespace NOA_Application6
{
    public class Program
    {
        public static void Main()
        {

            Debug.Print("Hello World! This is NOA.");
            Ton NewTon = new Ton("DO");
            NewTon.PrintTon();
```

```
            Thread.Sleep(Timeout.Infinite);
        }
    }

    class Ton {  // Ton בשם מחלקה פתיחת

        string _Ton;  // המחלקה של משתנה הגדרת


    public Ton (string Bt){  // של בנאי יצירת Ton המחלקה משתני את שיאתכל המשתנה והגדרת
המחלקה

        _Ton = Bt; //  Bt הבנאי של במשתנה מאותחל _Ton המחלקה של המשתנה
            }

    public void PrintTon()
    {

        Debug.Print("the ton is" );
        Debug.Print(_Ton);
    }

    }

}
```

עכשיו אנחנו נרצה להוסיף גם את הצליל של התו

טבלת תווים ותדרים (תדרים נמדדים ב-Hz)

תו: \ אוקטאבה:	0	1	2	3	4	5	6	7	8
C דו	16.3516	32.7032	65.4064	130.813	261.626	523.251	1046.50	2093.00	4186.01
C#	17.3239	34.6478	69.2957	138.591	277.183	554.365	1108.73	2217.46	4434.92
D רה	18.3540	36.7081	73.4162	146.832	293.665	587.330	1174.66	2349.32	4698.64
D#	19.4454	38.8909	77.7817	155.563	311.127	622.254	1244.51	2489.02	4978.03
E מי	20.6017	41.2034	82.4069	164.814	329.628	659.255	1318.51	2637.02	5274.04
F פה	21.8268	43.6536	83.3071	174.614	349.228	698.456	1696.91	2793.83	5587.65
F#	23.1247	46.2493	92.4986	184.997	369.994	739.989	1479.98	2959.96	5919.91
G סול	24.4997	48.9994	97.9989	195.998	391.995	783.991	1567.98	3135.96	6271.93
G#	25.9565	51.9131	103.826	207.652	415.305	830.609	1661.22	3322.44	6644.88
A לה	27.5	55.0	110.0	220.0	440.0	880.0	1760.0	3520.0	7040.0
A#	29.1352	58.2705	116.541	233.082	466.164	932.328	1864.66	3729.31	7458.62
B סי	30.8671	63.7354	123.471	246.942	493.883	987.767	1975.53	3951.07	7902.13

עכשיו נוסיף לתוכנית הראשית את התוים ונבדוק שהם נשמעים טוב.

ראשית נוסיף את הסיפריה של PWM.

```
using System;
using Microsoft.SPOT;
```

```
using Microsoft.SPOT.Hardware;
using Stm32;
using System.Threading;
using System.Text;

namespace NOA_Application6
{
    public class Program
    {
        static PWM spk = new PWM(Cpu.PWMChannel.PWM_6, 1046, 0.5, false);

        public static void Main()
        {


            Debug.Print("Hello World! This is NOA.");
            Ton NewTon = new Ton("DO");
            spk.Start();
            Thread.Sleep(1000);
            spk.Stop();
            spk.Frequency = 1174;
            spk.Start();
            Thread.Sleep(1000);
            spk.Stop();

        }
    }


    class Ton
    {  // Ton בשם מחלקה פתיחת

        string _Ton;  // המחלקה של משתנה הגדרת


        public Ton(string Bt)
        {  // המחלקה של בנאי יצירת Ton המחלקה משתני את שיאתכל המשתנה והגדרת

            _Ton = Bt; //  Bt הבנאי של במשתנה מאותחל _Ton המחלקה של המשתנה
        }
```

```
    }
}
```

הוסף את שאר התווים לפי התדירות.

המשך כתיבת מחלקות :

איך מגדירים מחלקה שבה אני לא יוצר שום בנאי (constructor) ?

נכתוב את הקוד הבא ונבין מה קורה :

```
using System;
using Microsoft.SPOT;
using Microsoft.SPOT.Hardware;
using Stm32;
using System.Threading;
using System.Text;

namespace NOA_Application6
{
    public class Program
    {
        public static void Main()
        {
            Debug.Print("Hello World! This is NOA.");
            Ton NewTon = new Ton("DO");
            NewTon.PrintTon();
            SpkTon NewSpkTon = new SpkTon();
            NewSpkTon.SoundSpkTon();
        }
    }


    class Ton
    {  // Ton בשם מחלקה פתיחת
        string _Ton;  // המחלקה של משתנה הגדרת
        public Ton(string Bt)
        {  // המחלקה של בנאי יצירת Ton המחלקה משתני את שיאתכל המשתנה והגדרת
```

```
            _Ton = Bt; //  Bt המחלקה של המשתנה _Ton הבנאי של במשתנה מאותחל
        }
        public void PrintTon()
        {
            Debug.Print("the ton is");
            Debug.Print(_Ton);
        }
    }
}

class SpkTon // המחלקה של משתנה הגדרת
{

    double _SpkTon;
    static PWM spk = new PWM(Cpu.PWMChannel.PWM_6, 1046, 0.5, false);

    public void SoundSpkTon()
    {
        Debug.Print("_SpkTon sound .");
        spk.Frequency = _SpkTon;
        spk.Start();
        Thread.Sleep(1000);
        spk.Stop();
    }

}
```

למרות שלא נוצר שום בנאי .

הקוד מאתחל את המשתנה של הבנאי בברירת מחדל :

במקרה של INT זה 0

במקרה של STRING זה NULL

במקרה של BOOL זה FALSE

אז

_SpkTon ב 0

כעת נוסיף לקוד בנאי נוסף עם פרמטרים :

ונרצה להפעיל את הבנאי ללא פרמטרים – לא נצליח .

ברגע שיש לנו בנאי עם פרמטרים לא נוכל להפעיל אותו ללא פרמטרים.

ולכן נהיה חייבים להכניס את המשתנים לתוך הבנאי : כך:

נשתמש במילה השמורה this

```
using System;
using Microsoft.SPOT;
```

```
using Microsoft.SPOT.Hardware;
using Stm32;
using System.Threading;
using System.Text;

namespace NOA_Application6
{
    public class Program
    {

        public static void Main()
        {
            Debug.Print("Hello World! This is NOA.");
            Ton NewTon = new Ton("DO");
            NewTon.PrintTon();
            SpkTon NewSpkTon = new SpkTon();
            NewSpkTon.SoundSpkTon();
            Thread.Sleep(1000);
        }
    }


    class Ton
    {  // Ton בשם מחלקה פתיחת
        string _Ton;  // המחלקה של משתנה הגדרת
        public Ton(string Bt)
        {  // המחלקה של בנאי יצירת Ton המחלקה משתני את שיאתכל המשתנה והגדרת

            _Ton = Bt; //  Bt הבנאי של במשתנה מאותחל _Ton המחלקה של המשתנה
        }
        public void PrintTon()
        {

            Debug.Print("the ton is");
            Debug.Print(_Ton);
        }

    }
}

class SpkTon // המחלקה של משתנה הגדרת
{
```

```
    double _SpkTon;
    static PWM spk = new PWM(Cpu.PWMChannel.PWM_6, 1046, 0.5, false);



    public SpkTon(double Frequency_HZ_SpkTon)
        {  // המחלקה של בנאי יצירת Ton המחלקה משתני את שיאתכל המשתנה והגדרת

            _SpkTon = Frequency_HZ_SpkTon; //  Bt מאותחל במשתנה של הבנאי _Ton
המחלקה של המשתנה
        }

    public SpkTon() : this(1108.73)
    {  // המחלקה של בנאי יצירת SpkTon המחלקה משתני את שיאתכל המשתנה והגדרת

    }

    public void SoundSpkTon()
    {
        Debug.Print("_SpkTon sound .");
        spk.Frequency = _SpkTon;
        spk.Start();
        Thread.Sleep(1000);
        spk.Stop();
    }

}
```

הערה :

ניתן לכתוב מספר בנאים שונים רק אם הפרמטרים שלהם שונים ואו מספרם.

משימות:

1.

רשום מחלקה בשם LedClass. עם שני constructor כאשר היא מדליקה לדים הבנאי הראשון בנאי ללא איתחול מגדיר את הלד האדום.

והבנאי השני מגדיר את הלד של הבנאי הראשון .

והפונקציה בפנים מדליקה את הלד למשך שניה ומכבה אותו.