

קריאה מקובץ (שימוש במחלקה קיימת)

עד היום הדרך לקבל נתונים בתוכנית הייתה לתת הודעה למשתמש מתוך התוכנית (על ידי הדפסה עם פעולת- Console.WriteLine - שהיא פעולה מתוך מחלקה שניקראת Console) ולאחר מכן ביצוע הקריאה בעזרת פעולה שניקראת: ReadLine שגם היא נימצאת בתוך המחלקה: Console.

לעיתים יש מצב שמקבלים כמות גדולה של נתונים על קובץ מחשב וזו תהיה טרחה גדולה שוב להקיש את הנתונים כאשר התוכנית שלנו מבקשת אותם.

לדוגמה אם לקוח של בנק מבצע כל מיני פעולות בחשבון שלו והפעולות נירשמות על גבי קובץ מחשב (הבנק חייב לתעד מה שהלקוח עושה בחשבון) ובסוף החודש רוצים להריץ תוכנית שאוספת את כל הפעולות ומסכמת אותן לדו"ח מסודר, התוכנית תרצה לקרוא מהקובץ שנישמר, במקום שאיזה פקיד שוב יזין את הנתונים של כל הלקוחות.

יש גם מצב שאפשר להוריד קובץ מהאינטרנט (נניח שמות של סרטים) ויש לנו תוכנית שממיינת את הקובץ לפי אורך סרט מהקצר לארוך ביותר, אז נצטרך להקליד לתוכנית עבור כל שורה בקובץ את שם הסרט ואורכו בדקות. אבל בקובץ שהורדנו יש 10,000 סרטים, אז אולי עדיף שהתוכנית עצמה תדע לקרוא מהקובץ את השמות ומשך הסרט במקום שנקליד בעצמנו?

אחד היתרונות של תוכנת מחשב היא עבודה עם נתונים רבים במהירות עצומה - נתונים רבים = קובץ.

ובכן, כפי שכבר התרגלנו לראות במקרים אחרים, שפת התכנות C# נותנת לנו כלים שחלקם נכתבו בעבר וניבדקו היטב, והם נימצאים בתוך מחלקות שאפשר ל"ייבא" לתוך התוכנית שלנו בעזרת משפט ה- using ולאחר מכן להשתמש בהם, אם נדע את שמותיהם והיכן הם נימצאים.

עבור העבודה עם קבצים, יש להשתמש בהוראה הבאה בראש התוכנית: `using System.IO`

השם IO זה קיצור של: Input / Output (או קלט/פלט)

בגוף התוכנית נישתמש במחלקה בשם: **StreamReader** שזה בעצם אומר: קורא רצף (stream) של תוים. כדי שבשלב הראשון יהיה לנו נוח לעבוד עם הקובץ, נחשוב על קובץ פשוט שבו יש לכל שורה משמעות (למשל על כל שורה נתונים של תלמיד בודד, או שכל שורה מייצגת פעולה בנקאית) - ולכן ניקרא את הקובץ שורה אחר שורה. כדי לעשות זאת, נגדיר מחרוזת שתכיל כל פעם שורה שניקראת מהקובץ:

```
string line;
```

השם המלא של הקובץ בדוגמה הבאה הוא:

```
c:\documents\file1.txt
```

(כונן c, תיקייה documents, ושם הקובץ הוא file1.txt - זה הקובץ שאתם תיצרו על הדיסק עם notepad)

וכעת בתוכנית ניצור את הקישור לקובץ (שהוא מחוץ לתוכנית) בעזרת ההוראה הבאה:

```
StreamReader sr = new StreamReader("c:\documents\file1.txt");
```

sr זה השם שבחרתי עבור מה שאקרא לו "הקובץ הלוגי", אפשר לתת כל שם חוקי של משתנה במקום sr אם תרצו (a, b, hal9000 וכד')

"c:\documents\file1.txt" - זהו הפרמטר שנתתי לפעולה הבונה של המחלקה StreamReader - זזהומה שאקרא לו "הקובץ הפיזי" שיושב אצלנו על הדיסק (כל אחד יכול ליצור קובץ משלו ולקרוא לו שם משלו ולשמור אותו במיקום שהוא יחליט).

אחרי שעשינו את הקישור בין הקובץ הפיזי והקובץ הלוגי (פעם אחת בתוכנית), אפשר להמשיך ולהשתמש רק בשם הלוגי של הקובץ, בהמשך התוכנית.

כעת נותר לנו לבצע את פעולת הקריאה מהקובץ ולתוך המשתנה בשם line שהגדרתי כמחרוזת. מדוע הגדרתי דווקא כמחרוזת ולא כשלם? כי בקובץ יכולים להיות כל מיני תווים, לא רק מספרים, ולכן מחרוזת יכולה להכיל כל רצף של תווים שהוא. אז הנה פעולת הקריאה:

```
line = sr.ReadLine();
```

נראה מוכר לא? כן אותה פעולה שבה השתמשנו לקרוא מהמסך, תשמש לנו לקריאה מהקובץ, וזיכרו שכעת המשתנה sr מייצג עבורנו את הקובץ שקישרנו אליו. הנקודה "מזמנת" את הפעולה על העצם המזוהה כ-sr (מיוג StreamReader)

פעולת הקריאה מזיזה גם את "הראש הקורא" לשורה הבאה בקובץ והוא נימצא כעת בהתחלת השורה הבאה. "הראש הקורא" זה כונון מגנטי על דיסק המחשב (נזכיר זאת בהמשך), שמסוגל לקרוא את הכתוב על הדיסק. כאשר קריאת הקובץ הסתיימה ומנסים לקרוא עוד שורה, הקריאה תתן ערך של null שהוא ערך ריק, ולכן נשתמש בלולאת while שתקרא מהקובץ שורה אחר שורה עד שניתקבל הערך null.

כעת נכתוב תוכנית קצרה שפשוט קוראת שורות מהקובץ ומדפיסה אותן על המסך:

```
using System;
using System.IO;
namespace FileApplication
{
    class Program
    {
        static void Main(string[] args)
        {
            StreamReader sr = new StreamReader("C:\documents\file1.txt");
            string line;
            line = sr.ReadLine();
            while (line != null)
            {
                Console.WriteLine(line);
                line = sr.ReadLine();
            }
        }
    }
}
```

אם שמתם לב בתוכנית הנ"ל, פקודת הקריאה: `line=sr.ReadLine;` נימצאת בשני מקומות. גם לפני התחלת לולאת ה-`while` וגם בתוך הלולאה.

הסיבה לטכניקה זו היא שאם הקובץ ריק, אין ברצוננו לנסות ולהדפיס את `line` (כי הוא לא יכיל דבר), ובצורה זו על ידי קריאת השורה הראשונה ואז ביצוע ה-`while` שבו יש תנאי שצריך להתקיים ש-`line` אינו `null` - מבטיח לנו שלא יהיה ניסיון הדפסה של `line` כי אם הקובץ ריק נקבל `null` ב-`line`

וכמובן, אחרי קריאת השורה הראשונה והדפסתה, אנחנו רוצים להמשיך ולקרוא את השורות הבאות בתוך הלולאה ואז ה-`while` שוב בודק אם לא הגענו לסוף הקובץ, שוב נדפיס ושוב ניקרא, כך עד לסיום הקובץ. לאחר קריאת השורה האחרונה בקובץ, שוב יהיה ניסיון קריאה, אבל הפעם `line` יקבל את הערך `null` וכך תסתיים הלולאה.

זו תוכנית מאד פשוטה שרק קוראת את כל השורות מהקובץ ומדפיסה אותן אחת אחרי השניה, אבל היא מדגימה לנו את שיטת העיבוד של הקובץ. במקום להדפיס את השורה, בעתיד, נוכל גם לכתוב משהו שיבצע פעולות שנרצה עם הנתונים של כל שורה (אולי סיכום מספרים, אולי חישובים שונים). הטכניקה הזו תהיה מסגרת העבודה עם קבצים.

תרגיל בית

להעתיק את התוכנית הנ"ל, ולשנות אותה בצורה כזו שהיא תדפיס כל שורה שנייה מהקובץ. לדוגמה אם הקובץ נראה כך:

line number 1
line number 2
line number 3
line number 4

התוכנית תראה על המסך:

line number 2
line number 4