סיכום נושא async,Task, parallel

1. אסינכרוני = לא בצורה סדרתית, כלומר פעולה יכולה להתחיל, אחרת יכולה להמשיך, ואז הראשונה תסתיים.

כשמדבירם על קוד אסינכרוני, מדברים על קוד שבינתיים עד שפעולה אחת מסתיימת – ממשיכים לפעולה נוספת

1. מקבילי- כמה גורמים אחראים לביצוע של כמ הפעלות. ומכיון שזה כמה גורמים – הפעולות תסיימנה יותר מהר.

ביצוע קוד אסינכרוני, וביצוע קוד מקבילי, החל מ .net 4 שניהם מתבצעים באמצעות Task.

(נמצא ב System.Threading.Tasks namespace)

איך יוצרים task?

1. כמו שיוצרים כל מופע, ב ctor שולחים ל task פונקציה (באמצעות delegate).
2. לזמן את הפונקציה Task.Run ולשלוח לה delegate. מאחורי הקלעים תיווצר task.
3. Task.Factory.StartNew…

כדי לפשט את התחביר של קוד אסינכרוני, החל מ .net 4.5 קבלנו את async await keywords.

פונקציה אסינכרונית:

1. מסומנת ב async keyword
2. מחזירה Task, ואם רוצה להחזיר ערך אזי Task<T>
3. לפי קונבנציה שמה – במצגת
4. בדרך כלל תהיה בה המתנה לאיזשהי פעולה – await – אחרת אין סיבה ליצור אותה async.
5. ניתן להגדיר async על פונקציה בלי שימוש ב await (לא הגיוני כל כך אבל ניתן), אך לא ניתן לבצע await בפונקציה שאינה async.

הפונקציה Wait מבצעת המתנה לסיום Task שאינו מחזיר ערך.

באופן כללי:

כאשר מבצעים await , הפונקציה הזו תפסיק, עד לקבלת תוצאה, ויוחזר הפיקוד לפונקציה הקוראת.

כאשר ה Main (והכוונה לכל תהליך ראשי) זימן task ולא המתין לסיומם, אזי אם התהליך הראשי הסתיים לפני סיום ה task, בברירת מחדל ה task לא תסתיים. ישנה אפשרות להגדיר במפורש שהתהליך ימתין עד לסיום ה task. לכ כדאי לוודא שממתינים לסיום כל מה שהתחילו.

מעקף – להמתין לקלט