**EXPRESS**

לא אלאה במורכבות של http (כדאי לקרוא קצת בויקיפדיה למשל), אבל דפי השרת:

* חייבים להשתמש בכל הקודים של האובייקט.
* חייבים לייבא אחד אחד את כל הקבצים (html, css, js) שנשתמש בהם
* ועוד ועוד....

זה קשה ומאריך מאוד מאוד את הקוד.

מודול express בא לפתור המון בעיות מסוג זה. (אמנם לא נכיר את הבעיות הללו כל כך, כדי שלא להיכנס למורכבות שבסוף נוותר עליה...)

סטנדרט של כל שימוש ב- node!

בחלק זה של הקורס נראה את הבסיס של express:

* יישום כללי
* יישום עם html
* יישום json

עם ההקלה של השימוש ב- express, נפתח לנו שער לדברים מורכבים ברמת הנתונים ולא ברמת ניהול השרת.

בגלל שלא התעמקנו יותר מדי במורכבות של כתיבה ללא express - הקובץ הזה לא מראה המון הקלות, אבל הוא אכן נותן את זה..!

**חובה להכין את הסביבה על ידי npm init -y והרצת npm i express ורצוי להתקין את nodemon**

1. ייבוא
   1. ה- express בדרך המוכרת
   2. כאן **מריצים** את מה שייבאנו. חובה לרשום סוגריים!. הערך המוחזר הוא יהיה האובייקט שאיתו נעשה את כל העבודה. תמיד נקרא לאובייקט זה בשם **app**
2. לאובייקט app זה יש הרבה שיטות, החשובות שבהן: get, post, put, delete, all, listen, ועוד אחת מיוחדת בשם use.   
   ה- get מיועד להקשבת השרת כאשר גולש מבקש לקבל משהו.   
   פרמטרים:
   1. ה- path לגביו הוא מבקש
   2. ה- callback – תגובת השרת לבקשה. הפרמטרים שלה הם ה- req,res שכבר הכרנו בעבר  
      בשורה זו אנו אומרים: כאשר בבקשת ה- get היה נתיב של / אז תבצע :
      1. log סתם בשביל הדיווח בטרמניל
      2. העיקר: תרשום מידע בדפדפן.   
         תוך כדי אנו יכולים לומר שה- http return code של פעולה זו היא 200, דהיינו תקין.
3. במידה וההרצה לא הסתיימה (על ידי send) – השרת יגיע לשורה זו, ומבצעת את get. עבור כל השיטות של app ועבור כל הכתובות - בצע שגיאה 404, והצג תוכן מתאים. (למשל: localhost:5000/about)
4. שיטה נוספת של app, דומה לדברים שלמדנו, המבצעת הקשבה לשרת

**משימונת כיתתית**: תנסה לשם התרגול לכתוב את הכל מחדש, מבלי להעתיק ב- copy-paste, והוסף טקסט עבור דף help של האתר. (כתיבה חוזרת ונשנית מאוד עוזרת להקניית החומר....)