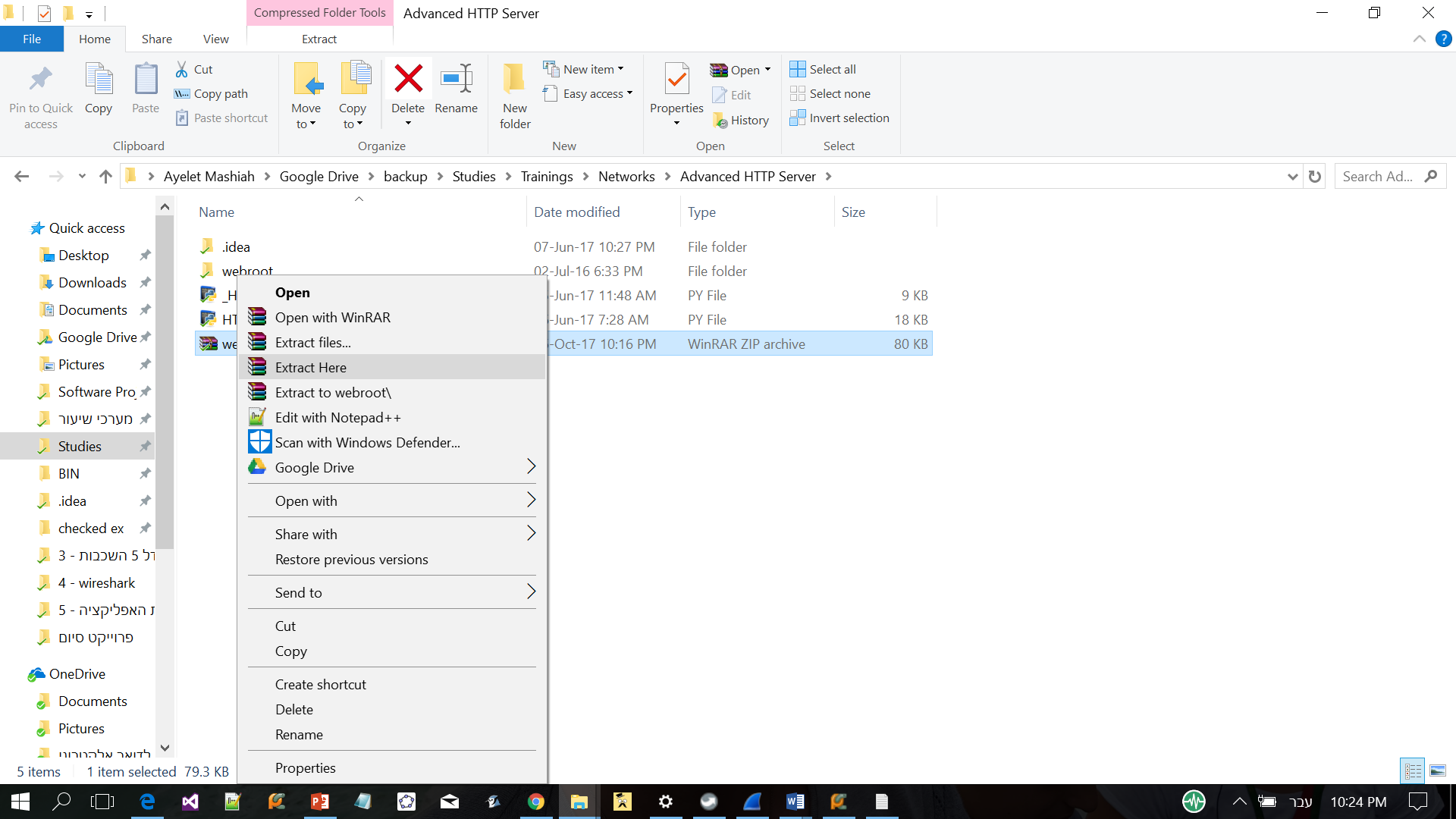
[](https://www.google.co.il/imgres?imgurl=https://blog.idrsolutions.com/wp-content/uploads/2017/05/http.jpg&imgrefurl=https://blog.idrsolutions.com/2017/05/why-http2-client-in-java-9-is-important/&docid=9akD0He2fkP-jM&tbnid=HFehawWVLOOVSM:&vet=10ahUKEwi1rqGA2NzWAhUCuBoKHZFnCK84ZBAzCBAoDjAO..i&w=596&h=480&bih=612&biw=1280&q=http%20client&ved=0ahUKEwi1rqGA2NzWAhUCuBoKHZFnCK84ZBAzCBAoDjAO&iact=mrc&uact=8)HTTP Server

# הקמת האתר הפיזי

לפני שניגש לכתיבת הקוד נקים את האתר הפיזי.

הורידו אל המחשבים שלכם את הקובץ webroot.zip מהמודל. שימו אותו בתיקייה ידועה על ה diskonkey או המחשב הנייד שלכם ופתחו אותו. לחצן ימני על העכבר 🡨 וביחרו Extract Here



# כתיבת ה server

כתיבת ה server היא מורכבת. על מנת להקל הוכנה עבורכם תבנית. לאחר שנוריד ונכיר את התבנית נבנה את ה server צעד אחר צעד

## תבנית ה Server

הורידו מהמודל את הקובץ HTTP\_server\_shell.py. פתחו את הקובץ ונסקור אותו

|  |  |
| --- | --- |
| main פתיחת socket ולולאה שמחכה לclients  כש client מתחבר, נקראת הפעולה handle\_client שעונה לבקשות  ה client עד לניתוק ה connection  שימו לב ל timeout. לפעמים הקשר בין השרת ללקוח נתקע וה timeout משחרר אותו. שימו לב שהוא גורם ל exception שבו יש לטפל |  |
|  |  |
| handle\_client טיפול בבקשות client עד שמגיעה בקשה שאינה תקינה  שימו לב שעליכם לקבל את ההודעות מהלקוח  אם התנתק ה connection נקבל מהלקוח ''  עבור כל הודעה הפעולה קוראת ל:   * validate\_http\_request * handle\_client\_request   אם הגיעה הודעה לא חוקית הקשר מנותק |  |
| validate\_http\_request פונקציה ריקה בה עליכם לבדוק את החוקיות של בקשת HTTP |  |
|  |  |
| handle\_client\_request לפעולה כמה חלקים:   1. טיפול ב default resource 2. טיפול בהזזה 3. בדיקת content type 4. קריאת ה data ע"י זימו הפעולה : get\_file\_data   השאירו את 2 הראשונים לסוף |  |

## כתיבת הקוד

נחלק את הכתיבה ל 5 שלבים:

1. ה server שלנו יחזיר את שם הקובץ בלבד.
2. ה server שלנו יחזיר את תוכן הקבצים
3. ה server שלנו יטפל בשגיאות
4. ה server שלנו יטפל בהזזה ששל קובץ - redirect
5. ה server שלנו יחזיר גם תוצאות של פעולות חישוביות

♥שימו לב! בסוף כל שלב עליכם לבדוק את ה Server שלכם בעזרת הדפדפן

### החזרת שם הקובץ בלבד

הקמת ה connection

* נקבע את ה**ip**  וה **port** של ה server. ה port של שרת האינטרנט שלנו כזכור לנו הוא תמיד **80**

**80**

היות ואתם רצים ל מחשב אחד ניתן לקבוע את ה IP להיות מחשב ה"בית" – '**127.0.0.1'**

שימו לב, **IP** ו **PORT** שניהם קבועים ולכן יש להגדירם בראש הקובץ באותיות גדולות.

* יש לנו קבוע נוסף, SOCKET\_TIMEOUT. בדקו באינטרנט אילו יחידות זמן הפעולה מקבלת (דקות, שניות , מילישניות) והגדירו את ה timeout לפרק זמן קצר
* כעת נשאר לנו לטפל בקבלת בקשות מהלקוח. הוסיפו לפעולה handle\_client את קבלת ההודעות מהלקוח. (השמשו ב client\_socket.recv)

טיפול בבקשה

לאחר שקיבלנו בקשה מהלקוח עלינו לטפל בה. נחלק את הטיפול ל 2 שלבים

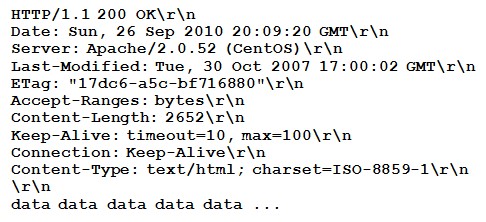
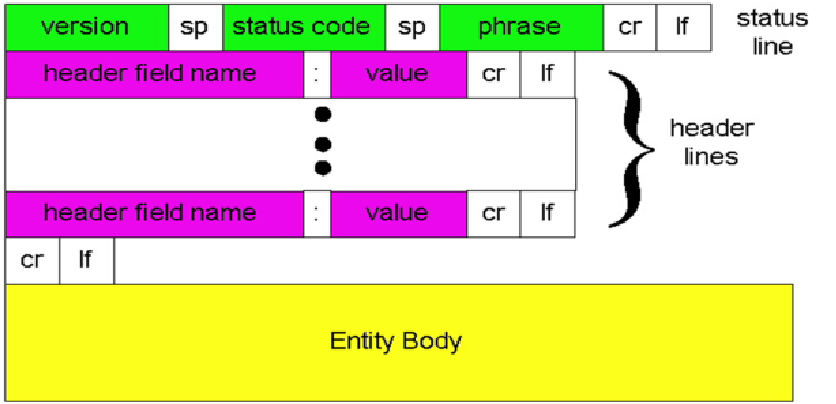
**בדיקת חוקיות** – ממשואת הפעולה validate\_http\_request כך שתבדוק את חוקיות הבקשה ותחזיר tuple . אם הבקשה חוקית הפעולה תחזיר (שם המשאב (True, ואם אינה חוקית תחזיר (None (False,

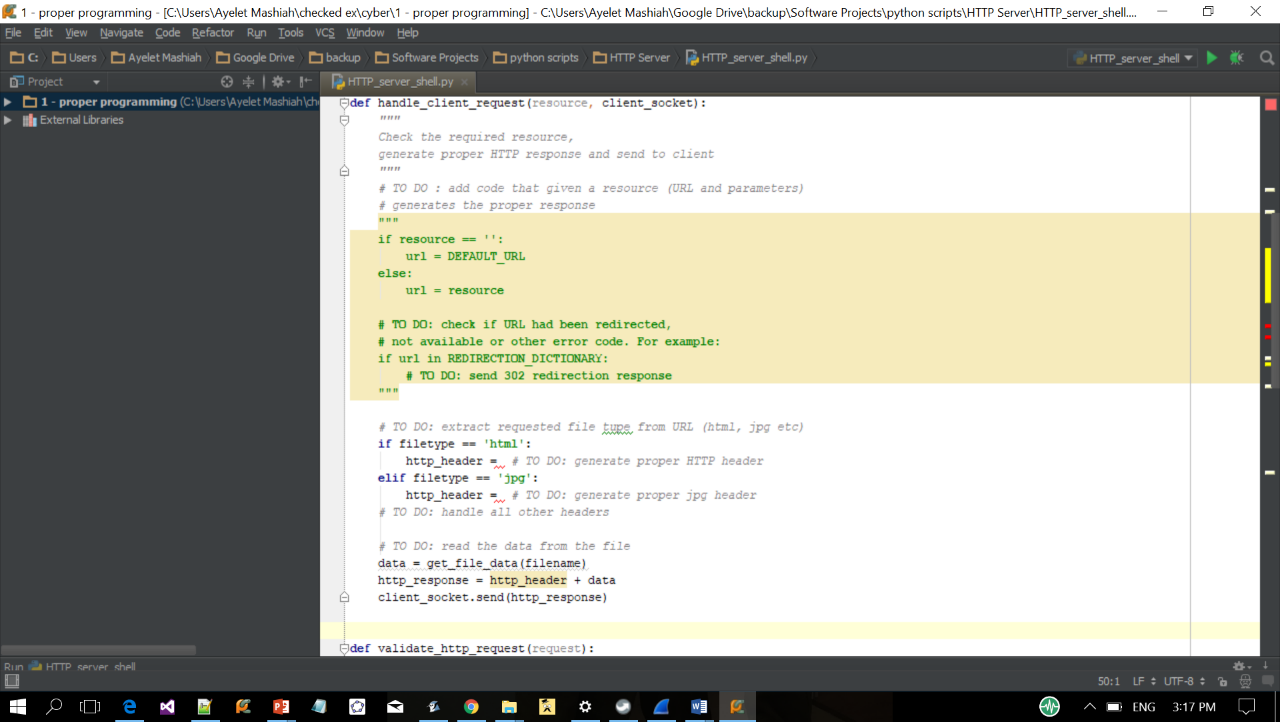
**בניית והחזרת תגובה**

**sp** - **מקרא:** רווח

**cr** - \r

**lf** - \n





בפעולה handle\_client\_request נשים בשלב זה את השורות הראשונות בהערה

אחריהן נבנה את התגובה http\_response

התגובה שלנו תכיל:

* HTTP/1.1 200 ok\r\n # שורת התגובה
* Content-Length: <data length> \r\n # אורך הנתונים המוחזרים
* Content-Type: <contrnt type> \r\n #שם הקובץ סוג הנתונים המוחזרים – ייקבע לפי
* \r\n
* Data # תוכן הקובץ

|  |  |
| --- | --- |
| קובץ | Content-Type |
| File.html, File.txt | 'text/html; charset=utf-8 |
| File.jpg, File.ico,  File.gif, File.png | image/jpeg |
| File.css | text/css |

בשלב זה נחזיר את שם הקובץ במקום את תוכנו. לכן, Conten-Length הוא אורך שם הקובץ ו Content-Type הוא 'text/html; charset=utf-8'

שם הקובץ: נגזור את שם המשאב מה URL וכדי לקבל את מיקום הקובץ נחבר אותו עם ה root. (webroot) לדוגמא אם התיקייה webroot נמצאת ב c:\cyber וקיבלנו בקשה ל: 127.0.0.1/imgs/ abstract.jpg  
הרי שהקובץ שלנו הוא:

c:\cyber\webroot\ imgs\ abstract.jpg

### החזרת תוכן הקובץ

נרצה להחזיר ללקוח את תוכן הקובץ שביקש.

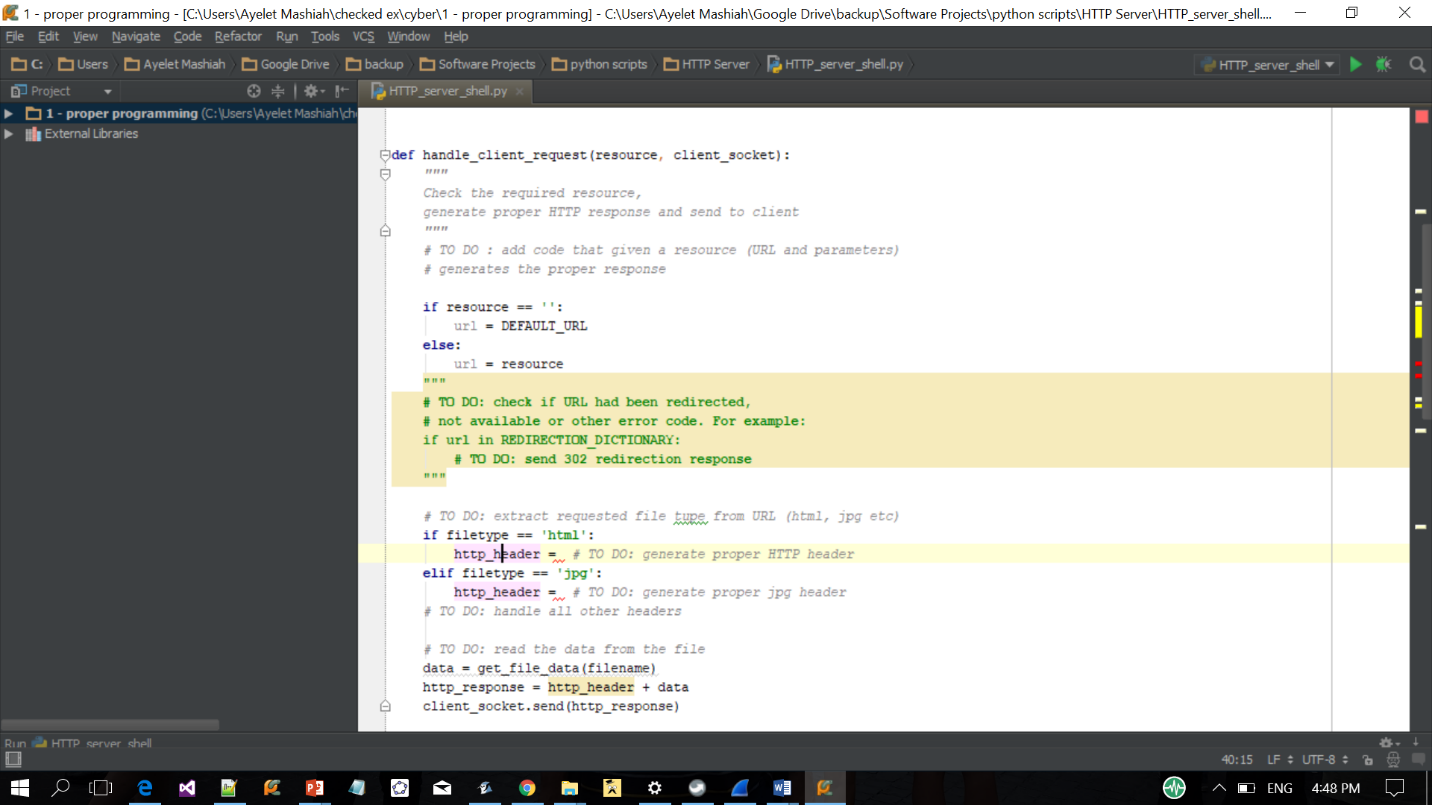
היזכרו בתרגיל Copy-paste בו העתקתם קובץ ממקום למקום. גם כאן עליכם לקרוא את תוכן הקובץ והפעם עליכם לשלוח אותו ללקוח . שימו לב שאתם צריכים לקרוא את הקובץ כיחידה אחת ולא בחלקים.

* ממשו את הפעולה get\_file\_data(filename) שתחזיר לכם את תוכן הקובץ אותו התבקשתם להחזיר
* ממשו פעולה המקבלת שם של קובץ ומחזירה את ה Content-Type שלו על פי הטבלה שלמעלה ובנו תגובה המכילה את תוכן הקובץ

### **החזרת index.html.**

הקובץ index.html הוא קובץ מיוחד. אם השרת מקבל בקשת GET ל-root (כלומר למיקום "/") - נחזיר את הקובץ index.html

הגדירו את הקבוע DEFAULT\_URL לדוגמא: DEFAULT\_URL = 'C:\\Cyber\\webroot\\index.html’.

פתחו את 4 השורות הראשונות בהערה שסגרנו בתחילת הפעולה handle\_client\_request

### טיפול בשגיאות

* החזירו בשורת התגובה 404 Not Found (במקום 200 OK) אם הקובץ לא קיים. אפשר ליבא את המודול os ולהשתמש בפעולה os.path.isfile
* החזירוError 500 Server Internal, במקום לנתק את ה connection – במקרה והבקשה נראית לכם לא חוקית

### Redirct

פעולת ה redirect היא פעולת הפניה מחדש שמבצע ה server. אם מגיעה בקשה למשאב ש"הוזז" כלומר שנמצא ברשימת ה"redirect" ה server מחזיר Moved Temporarily 302 בשורת התגובה וב header נוסף השדה location שערכו המיקום הנכון של הקובץ מתחת ל root. הclient לאחר קבלת תשובה כזאת, מבקש מה server את המשאב במיקום הנכון

הוסיפו REDIRECTION\_DICTIONARY שמכיל את שמות המשאבים ש"הוזזו" ואת מיקומם החדש

לדוגמא:

REDIRECTION\_DICTIONARY = {'/js/box1.js':'/js/box.js'}

במילון שלנו כתוב שהקובץ box1.js עבר ל box.js.

כעת כשתגיע בקשה לקובץ נבדוק האם הוא "הוזז" כלומר האם הוא נמצא ב dictionary ונחזיר header מתאים

### פעולות חישוביות

נגדיר בשרת שלנו 2 פעולות

* **calculate-next?num=x –** כשנקבל פנייה למשאב calculate-next נחזיר את x+1

לדוגמא: עבור http://127.0.0.1/calculate-next?num=11 יוחזר 12

♥נשים לב שה content-type במקרה הזה הוא text/plain

* **calculate-area?height=h&width=w - –** כשנקבל פנייה למשאב calculate-area נחזיר את חישוב השטח של משולש (w \* h) /2, על סמך שני פרמטרים: גובה המשולש והרוחב שלו.   
  לדוגמא: עבור http://127.0.0.1/calculate-area?height=3&width=2 יוחזר 3\*2/2 כלומר "3.0"

# בהצלחה