פתרון לתרגיל HTML עם API

התרגיל: עליכם לכתוב תוכנית שמציגה בדפדפן לוח טיסות המראות ונחיתות מעודכן. הנתונים יעודכנו דרך API:

https://data.gov.il/api/3/action/datastore\_search?resource\_id=e83f763b-b7d7-479e-b172-ae981ddc6de5&limit=15

.

### שלב א’ בפתרון: לחזור על מה שלמדנו בכל אחד מהתחומים בפייתון, SQL ו HTML:

* **תחום פייתון:** 
  + - למדנו איך לפתח class שמתאים לטבלה של נתונים.
    - מכיוון שמדובר בטיסות, ה class המתאים הוא טיסה, Flight.
    - למדנו איך לאתחל את ה class עם תכונות (properties) שמתאימות לשדות של הטבלה שמעניינות אותנו: חברה, תאריך חברה, יעד וכו’.
    - למדנו איך ליצור אובייקט מתוך ה class. באובייקט הזה נאחסן נתונים של טיסה אחת. אם רוצים לבצע פעולות כגון שמירה של האובייקט כשורה בטבלה html, הגיוני להוסיף ל class מתודה (פונקציה) שעושה זאת, כך שנוכל להפעיל את הפונקציה על האובייקט בלולאת for.
    - למדנו איך מייבאים את ספריית sqlite3 ואיך ניגשים ל database ומריצים עליו שאילתות ב SQL.
    - למדנו ליצור HTML ולשמור אותו בקובץ.
* **תחום SQL** 
  + - למדנו איך יוצרים טבלה עם CREATE
    - למדנו איך שומרים שורה בטבלה עם INSERT
    - למדנו בשיעורים קודמים איך שולפים שורות מהטבלה עם SELECT
* **תחום HTML** 
  + - למדנו ליצור טבלת HTML ולמלא אותה בנתונים.
    - למדנו בשיעורים קודמים איך להוסיף עיצוב עם CSS.

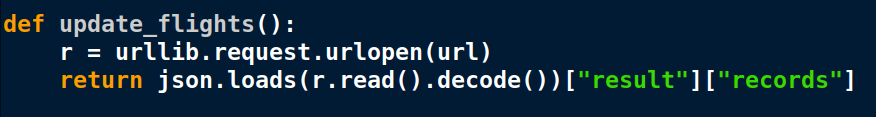
### שלב ב’ בפתרון:

#### משימה 1: יש להוריד את לוח הטיסות דרך API כל 15 דקות

פה בעצם אין שום דבר חדש, רק שילוב של דברים שלמדנו:

* שימוש ב urllib
* שימוש ב json

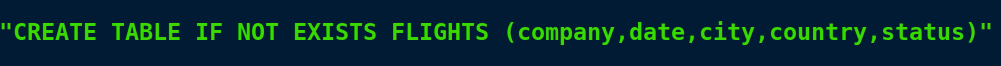
צריך לכתוב פונקציה שמורידה את ה JSON מה API ומחזירה אותו כמילון.



#### משימה 2: לשמור את הטיסות ב database עם SQL

פה צריך לחזור לחלק של ה SQL שסיכמנו בהכנה (ראו למעלה חלק א’ של הפתרון).

קודם כל צריך ליצור טבלת טיסות עם הנתונים המתאימים:

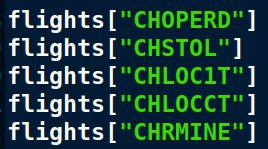


אחרי שיצרנו את הטבלה, אנחנו צריכים לשמור שם את הטיסות. למדנו איך שומרים שורה אחת עם INSERT.

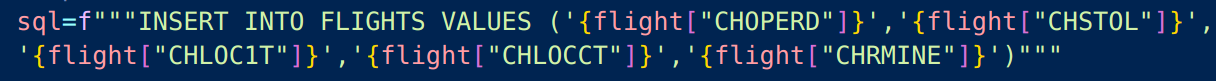
מכיוון שצריך לשמור הרבה שורות, הגיוני לעשות זאת בלולאת for.

איך נהפוך את רשימת הטיסות שקיבלנו מה API לשורות ב SQL?

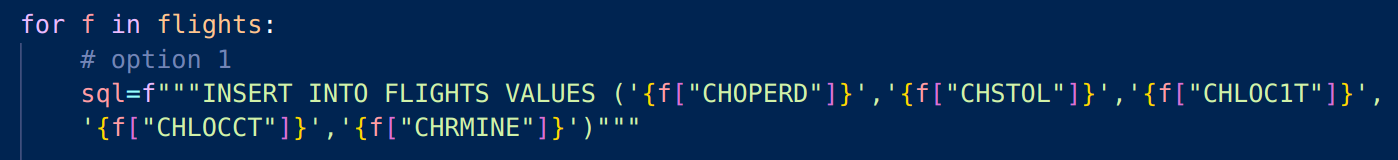
1. ברשימה יש מילונים, כשכל מילון ברשימה הוא בעצם טיסה. כלומר, כל מילון ברשימה שקיבלנו מהפונקציה update\_flights הוא שורה בטבלה.
2. צריך לחלץ מהמילון הנ"ל את נתוני הטיסה עצמם שמעניינים אותנו, כלומר חברה, תאריך, עיר, ארץ וסטטוס.
3. אם נסתכל במילון של טיסה אחת נגלה שהמפתחות המתאימים הם:

****

אלה הם הנתונים בטבלה. צריך רק להכניס אותם לשאילתת SQL INSERT מתאימה:

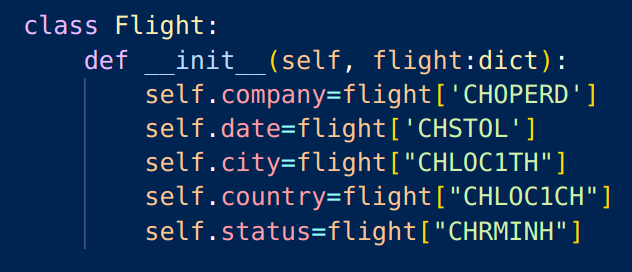
****

ואז לבצע את ה INSERT על כל הרשימה בלולאת for:

****

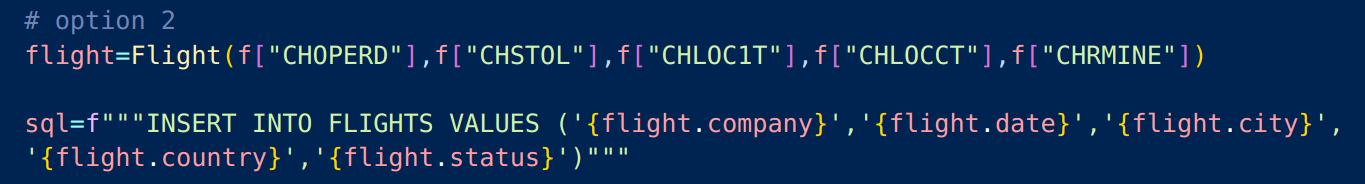
נשים לב שהשאילתה היא מסורבלת, ארוכה וקשה לקריאה.

היינו רוצים להתייחס למשתנים בצורה יותר טובה, למשל flight.company. נעשה זאת בעזרת class:



ה class מאפשר לנו ליצור אובייקט מסוג Flight, ולאתחל אותו עם הערכים של כל טיסה ברשימה. הטיסה היא מילון, כיצד ניצור אובייקט Flight ממילון?

אפשרות 1: נכניס את הערכים ידנית, כלומר:

****

זה עדיין מסורבל, למרות שהשאילתה נראית יותר קריאה.

איך אפשר לשפר את הקוד כך שנוכל לאתחל את האובייקט וליצור את השאילתה בקלות?

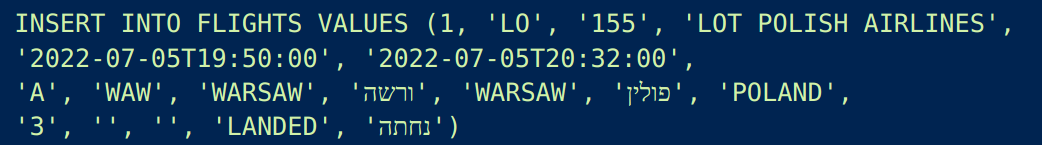
נזכור שמילון הוא בעצם keys ו values, ואפשר לקבל בקלות את ה values. בשביל השאילתה אנחנו צריכים string שנראה כך:



**נשים לב שזהו בדיוק מבנה של tuple.** כלומר אם ניקח את ה values של המילון ונמיר אותם ל tuple, ואז נמיר את ה tuple ל string, נקבל את מה שרצינו:



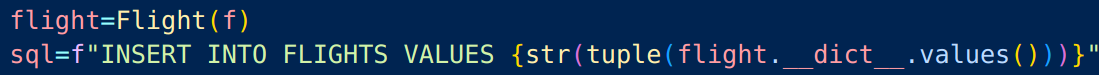
הבעיה כאן היא שנקבל את כל השדות של הטבלה מה API, כולל ערכים שלא רצינו. נסו זאת:



כדי לסנן את הערכים נשתמש ב class שיצרנו, Flight. מכיוון שה class מאותחל רק עם הערכים שרצינו, אפשר לאתחל אותו ישירות עם המילון (ערכי הטיסה):



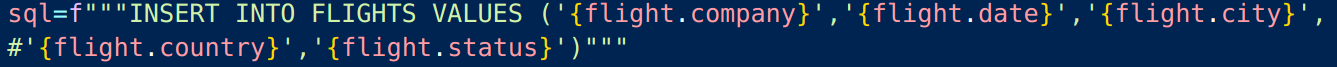
ואז לקחת את "מילון הקסם" של האובייקט flight, שאוטומטית נותן את התכונות שאנחנו רוצים מתוך האובייקט:



ועכשיו קיבלנו את השאילתה שרצינו באופן אוטומטי:



במידה וזה מבלבל, או שרוצים לשלוט ידנית בתכונות שמכניסים, אפשר להשתמש ב SQL של האפשרות הראשונה, עכשיו שאנחנו יודעים לאתחל את האובייקט באופן אוטומטי:

החסרון המאד משמעותי בנוסח זה הוא שנצטרך לעדכן ידנית את השאילתה בכל פעם שה class משתנה.

### משימה 3: להציג את הטיסות ב HTML

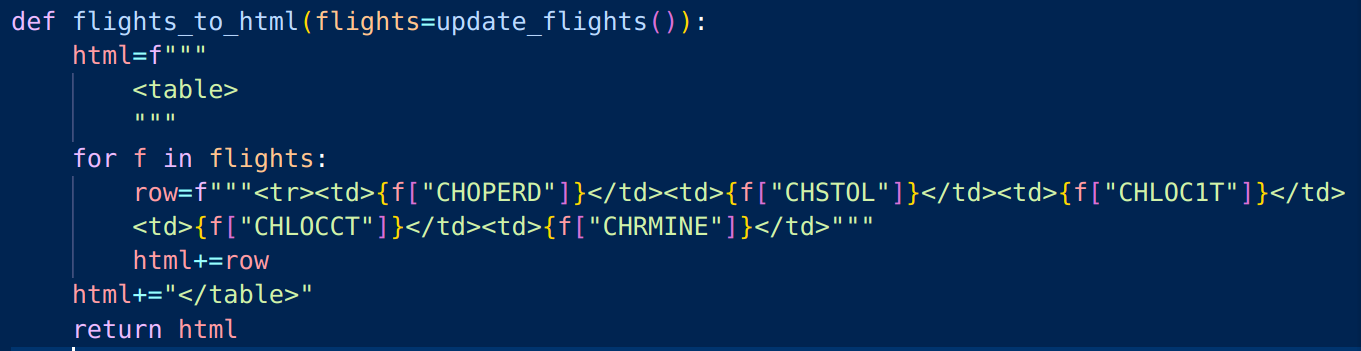
יש שתי אפשרויות: לקחת את הטיסות ישירות מה API ולהציג אותן, או לקחת אותן מה database. נכתוב פונקציה שלוקחת ישירות מה API ובשיעורים הבאים נראה איך לוקחים מה database.

יש לנו כבר רשימת טיסות שמקבלים מהפונקציה update\_flights. בעצם צריך לעבור על כל טיסה ברשימה ולכתוב אותה לתוך שורה בטבלת HTML.

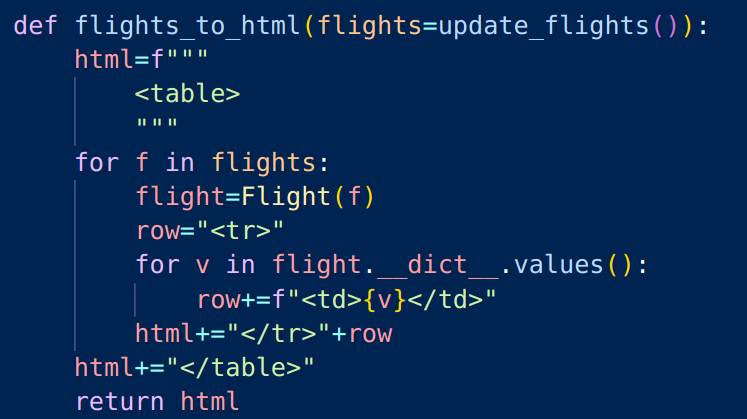
הגיוני לעשות זאת בלולאת for. כל איטרציה בלולאה תבנה שורת HTML חדשה כלומר:



אפשר לעשות זאת ידנית כפי שעשינו ב SQL:



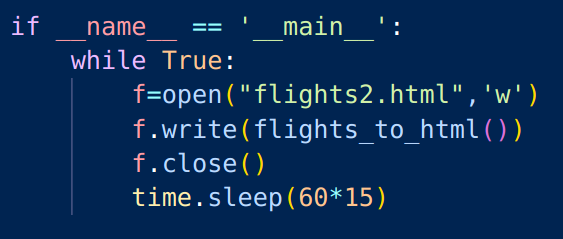
בעצם, זאת בעיה מאד דומה ליצירת שורה בטבלה עם SQL, רק עם HTML! ננסה להשתמש באותו פתרון:



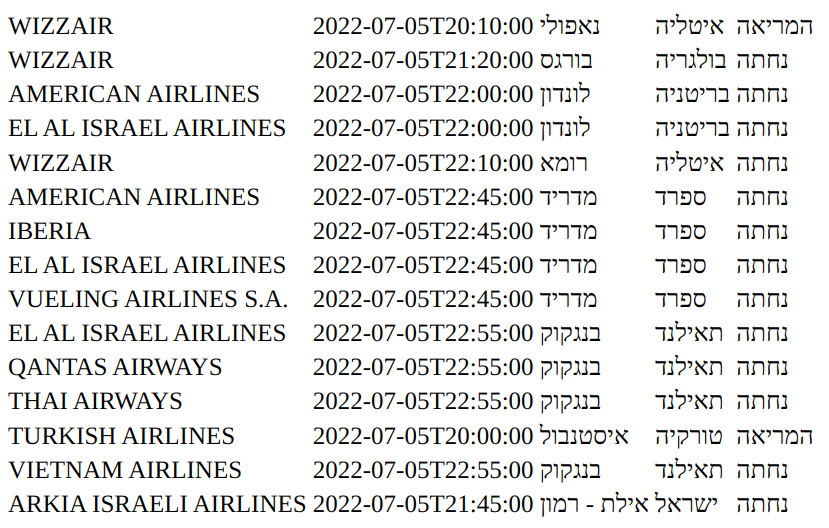
הפתרון הזה עובד ונותן את אותם יתרונות כמו ב SQL.

עכשיו כשיש לנו את ה HTML צריך לשמור אותו בקובץ, ולעדכן כל רבע שעה.

נשתמש בלולאת while ונקרא לפונקציה time.sleep כדי לחכות 15 דקות לפני העדכון הבא:



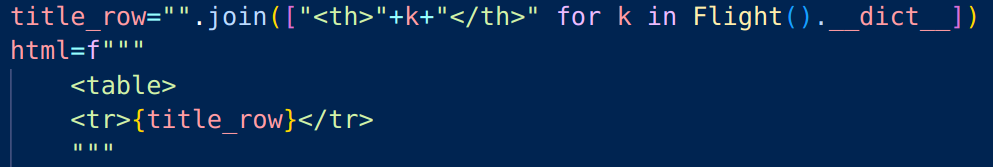
כך נראה דף הווב שלנו עד כה:

חסרה שורת הכותרת, וקצת עיצוב.

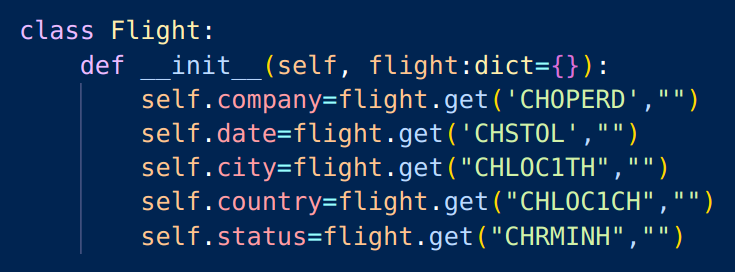
את שורת הכותרת אפשר להוסיף ידנית:



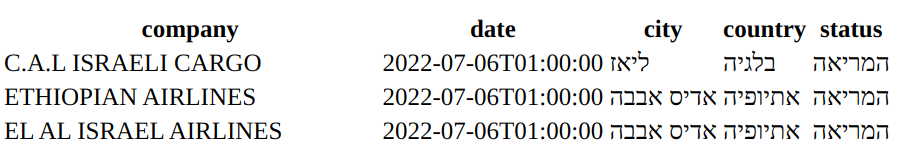
או באופן אוטומטי ע"י ה keys של האובייקט flight עם הטריק הבא list comprehension:



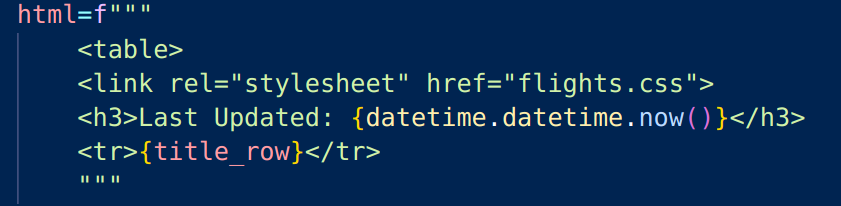
שימו לב שהשתמשנו ב ()Flight, כלומר אובייקט המאותחל בלי פרמטרים. כדי שנוכל לקבל אובייקט ()Flight כזה שינינו את ה \_\_init\_\_ של Flight והשתמשנו ב get של מילון:

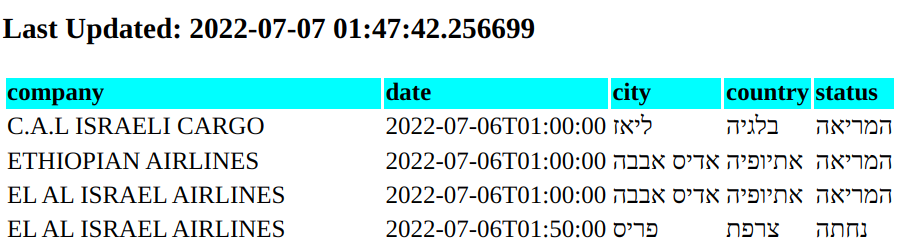


הקוד הנ"ל נותן לנו את שורת הכותרת:

****

נוסיף עיצוב ואת שעת העדכון:





כדי להבין כמה חשוב לעבוד עם OOP נראה מה קורה עם רוצים להוסיף עוד עמודה לטבלה, למשל את שם העיר באנגלית.

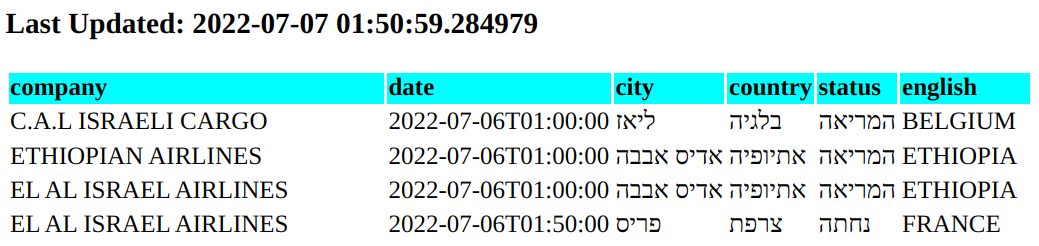
כדי להכניס את השינוי הזה צריך:

* לעדכן את טבלת ה SQL עם העמודה החדשה
* לעדכן את שאילתת ה SQL INSERT
* לעדכן את טבלת ה HTML

נוסיף עוד שדה אחד ל class:

****

השינוי האחד הזה מספיק כדי להוסיף את העמודה החדשה לטבלת ה HTML, וגם להוסיף את העמודה החדשה לשאילתת ה SQL INSERT:

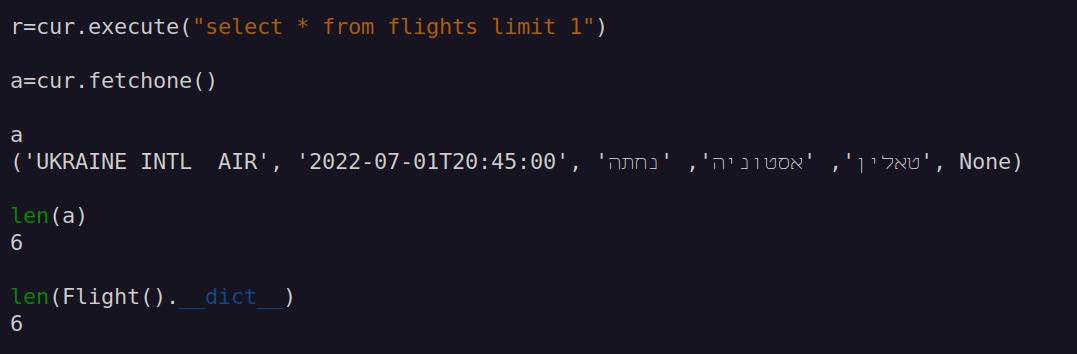
****

****

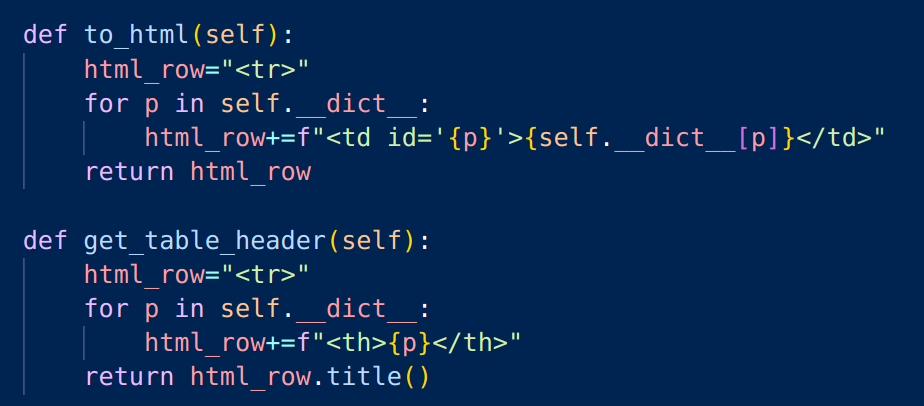
נשאר רק להוסיף את העמודה לטבלה עצמה, זהו קוד שמריצים פעם אחת (בדרך כלל במודול חיצוני) ע"י ה SQL הבא:



בונוס: אפשר כמובן ליישם לוגיקה שתזהה אם ה class השתנה ותוסיף את העמודה ב SQL אוטומטית,למשל ע”י השוואה בין מספר העמודות בטבלה לבין מספר התכונות במחלקה:



אפשר גם להכניס את יצירת שורת ה HTML ל class עצמו (כאן הוספנו גם את ה id של טבלת ה HTML שיעזור לנו בשלב ה javascript):

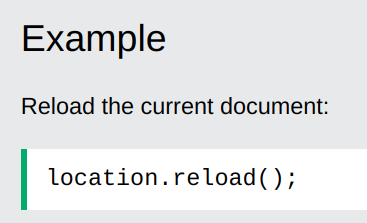


### משימה 4: refresh דרך javascript

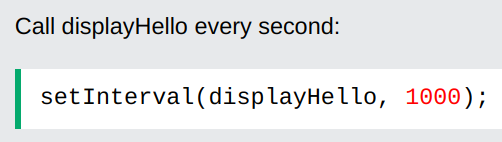
בשלב הזה אנחנו עובדים עם הדפדפן בלבד, כלומר יש לנו את הדף HTML מוכן וטעון בדפדפן, ומתוכו טוענים את קובץ ה js:

כל הקוד javascript שנכתוב ב flights.js ירוץ כשטוענים את הדף HTML.

שימו לב שקיבלנו רמז: להשתמש בפונקציה location.reload. נחפש בגוגל את הפונקציה הזאת:



זהו התחביר של הפונקציה. אבל, עדיין צריך להריץ אותה כל פרק זמן מסויים. נחפש בגוגל "js time interval”:



כלומר, setInterval מקבלת כפרמטר את הפונקציה שיש להריץ, ואת הזמן שיש לחכות באלפיות שניה. אם כך הקוד שלנו צריך להיות:



כדי להוסיף את השעה בה בוצע הריפרש, צריך להשתמש ב-()Date:



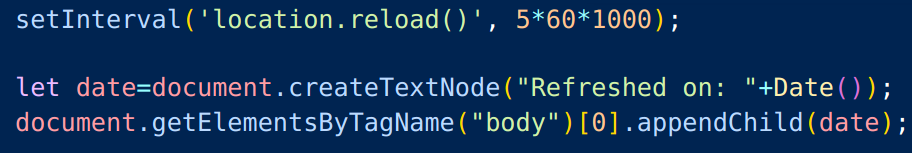
איך נכניס את התאריך לתוך ה HTML? נוסיף רכיב HTML חדש דרך javascript, ונכתוב לתוכו את התאריך. כל פעם שהדף יתרפרש, התאריך יתעדכן.

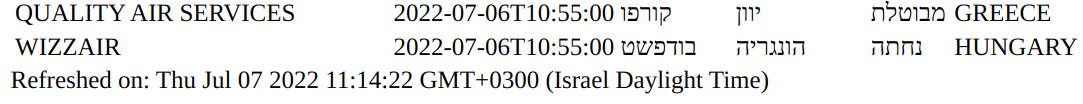
נשתמש בפונקציה document.createTextNode שיוצרת אלמנט חדש מסוג טקסט עם התאריך והשעה הנוכחיים:

 כדי להוסיף את האלמנט החדש date שיצרנו ל body, נשתמש בקוד הבא:

מכיוון שלא נתנו id ל body, אנחנו מקבלים אותו דרך getElementsByTagName שנותן לנו מערך של כל ה tags מהסוג body. מכיוון שיש רק אחד, ניקח את האינדקס 0 ונוסיף לו את האלמנט date עם appendChild.

לסיכום, הקובץ flights.js שלנו:

והדף ווב עם התאריך:



### משימה 5: שנו אוטומטית את הצבע של שורה עם טיסה מבוטלת לאדום (או צבע אחר)

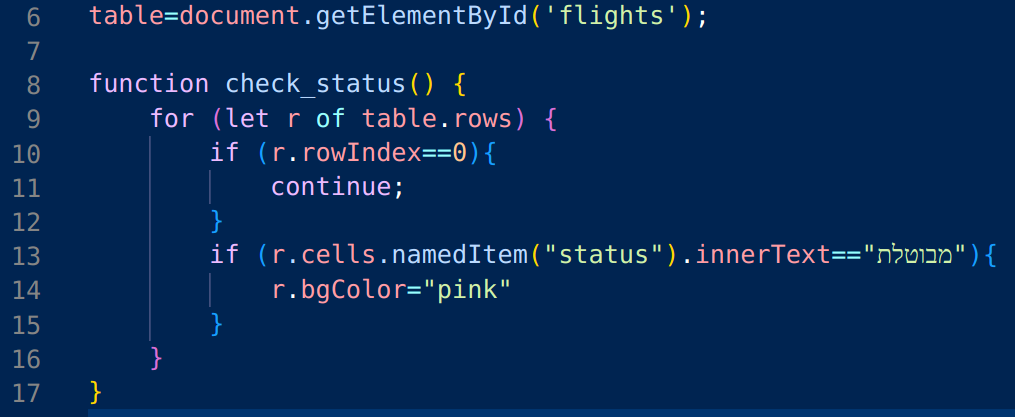
מהו האלגוריתם:

* 1. לעבור על כל רשימת הטיסות אחת אחת, כלומר על השורות בטבלה
* 2. בכל מעבר, לבדוק עם הטיסה מבוטלת לפי התא המתאים בשורה (status)
* 3. אם הטיסה מבוטלת, לשנות את צבע הרקע של השורה

זה נראה כמו משימה שמתאימה ללולאת for. אכן יש ב javascript לולאותfor דומות לפייתון, עם תחביר קצת שונה.

איך נשיג את רשימת השורות בטבלה? נשתמש בתכונה rows של table ב javascript. כדי להקל על עצמנו, נוסיף id לטבלה:

הקוד שנוסיף ל flights.js:



הסבר:

שורה 6: שמים את האובייקט table במשתנה.

שורה 9: לולאת for ב JS, שימו לב לסוגריים.

שורה 10: דילוג על שורת הכותרת בעזרת אינדקס 0 של מערך השורות rows

שורה 13: שימוש ב namedItem של אובייקט cells. זהו בעצם מילון עם המפתחות שמות התאים. שימוש ב innerText על אלמנט כמו תא בטבלה יתן לנו את התוכן הפנימי שלו, כלומר את ה string שאנחנו מחפשים.

שורה 14: שינוי הצבע של כל השורה r עם bgColor.

### משימה 6: הוסיפו חיפוש (נניח לפי שם חברה) בטבלה דרך JS.

את ה HTML של תיבת החיפוש יש להוסיף רק ב JS (document.createElement)

כדי לעשות include לקובץ javascript מתוך HTML:

<script src='flights.js'></script>