

## תדריך: אינטרנט של הדברים

אינטרנט של הדברים הינו חיבור של דברים, שבדרך כלל הינם חיישנים "טפשים" שביכולתם רק לאסוף מידע, לרשת האינטרנט וכך ניתן לעבד את המידע, לקשר בין "דברים" שונים, ולהפעיל "דברים" שונים.

### דוגמאות:

- המטריה דרך האינטרנט מגלה שמזג האוויר צפוי להיות גשום, והיא מתחילה להבהב.
- המקרר סורק את תכולתו, ומגלה שחסר חלב, ואז תאריך תפוגתו עבר, מזמין דרך האינטרנט חלב חדש.

כדי לחבר חיישנים טפשים לאינטרנט צריכים קודם כל לחבר אותם לבקר שבאמצעותו מיישמים את פרוטוקול התקשורת (TcpIp) בד"כ משתמשים בארדואינו Arduino או ברסברי פי Raspberry או בביגלבורד BeagleBoard, ותהליך זה דורש ידע בחשמל, וחיווט בנוסף לקידוד קוד.

לכן, ה"דברים" שנשתמש במעבדה הינם חיישני הרובולגו, שכבר מחוברים לבקר. אנו נחבר את החיישנים באמצעות בלוטוס למחשב, ומהמחשב נתחבר לאינטרנט. בקיצור אנחנו נבנה מערכת שדרך האינטרנט נוכל בכל מקום בעולם לקרוא את ערכי החיישנים של הרובולגו ולהפעיל אותו.

### מבנה המערכת פיזית:

- "מחשב קרוב" (במעבדה נממש אותו **במערכת ההפעלה חלונות**) שיהיה קרוב פיזית לבקר הרובולגו, יתקשר איתו באמצעות בלוטוס ויבצע את המשימות הבאות:
  1. יקרא את ערכי החיישנים מהרובולגו, ויכתוב אותם למסד הנתונים.
  2. יקרא את הוראות המשתמש ממסד הנתונים, "ויכתוב" אותם לרובולגו.
- "מחשב רחוק" (במעבדה נממש אותו **במכונה וירטואלית במערכת ההפעלה לינוקס**) אבל הוא יכול להיות בכל מקום שבו יש אינטרנט. שיבצע את המשימות הבאות:
  1. יקרא את ערכי החיישנים ממסד הנתונים, ויצג אותם למשתמש.
  2. יכתוב את הוראות המשתמש למסד הנתונים.

### חבילות תוכנה וסביבות עבודה:

- נעבוד בסביבת לינוקס (מחשב רחוק) וחלונות (מחשב קרוב), ונקודד בפייטון.
- לקישור בין הרובולגו למחשב נשתמש בפייטון NXT כמו שעשינו בתדריך אנדרואיד.
- במסד הנתונים נשתמש ב-REDIS, זהו מסד נתונים מסוג מפתח וערך (key value).
- לסביבת הפיתוח האינטרנטית נשתמש בקיווי, ובחלקה השני של המעבדה ב-Dash.

### סרטי עזרה:

- [מכונה וירטואלית.](#)
- [לינוקס.](#)
- [תאוריה IOT.](#)

עודכן ע"י גד הלוי 02/24 כל הזכויות שמורות לאוניברסיטת ת"א ולמחבר.  
אין להשתמש בתדריך זה ואז בחלקים ממנו ללא אישור המחבר.  
עמוד 1 מתוך 8

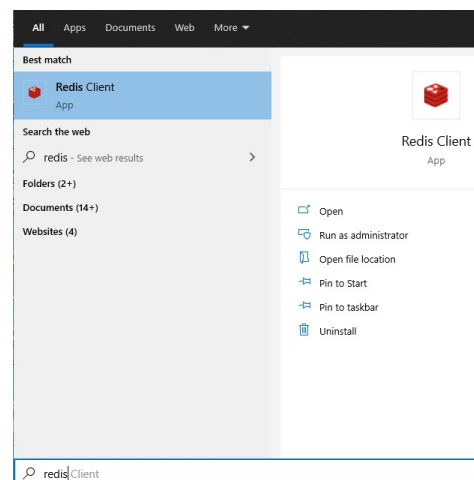
### קבצי עזרה בורפייר:

- IOT1 קובץ עזרה פייטון לשרת.
- IOT2 קובץ עזרה פייטון לממשק קיוי.
- IOT3 בדיקת לקוח.
- IOT4 בדיקת שרת.
- IOT5 קובץ עזרה ב-Dash.

### חיישן הצבעים של הרובולגו:

צבע	ערך
שחור	1
כחול	2
ירוק	3
צהוב	4
אדום	5
לבן	6

### הפעלת תוכנת רדיס במערכת הפעלה חלונות:



הדפס redis Client בשורת החיפוש בחלונות, ופתח את התכנה. החלון להלן צריך להופיע.

```

Redis Client
redis 127.0.0.1:6379> set Ohazan4:kivun Forward
OK
redis 127.0.0.1:6379> set Ohazan4:kivun Stop
OK
redis 127.0.0.1:6379> set Ohazan4:kivun Back
OK
redis 127.0.0.1:6379> set Ohazan4:kivun Stop
OK
redis 127.0.0.1:6379> set Ohazan4:kivun Exit
OK
redis 127.0.0.1:6379> get Ohazan4:kivun
"die"
redis 127.0.0.1:6379> _

```

עודכן ע"י גד הלוי 02/24 כל הזכויות שמורות לאוניברסיטת ת"א ולמחבר.  
 אין להשתמש בתדריך זה ואז בחלקים ממנו ללא אישור המחבר.  
 עמוד 2 מתוך 8

### המשימה בגדול:

המשתמש במחשב הרחוק ינהג את הרובולגו, ע"ס הכיוון שיבחר הרובולגו יסע. בנוסף המשתמש יוכל לקרוא את ערך הצבע שחיישן הרובולגו חש במעבדה במרחק אלפי קילומטרים מהמחשב הרחוק.

### חלק ראשון:

- במשימה זאת יש לבנות את תוכנת המחשב הקרוב ללא ממשק משתמש, ואת תוכנת המחשב הרחוק עם ממשק משתמש שימומש בקיוי.
- יש להשתמש במטלה הראשונה שכתבתם במעבדת אנדרואיד כבסיס.
  - במערכת הפעלה חלונות, יש לפתוח פרוייקט חדש ב-pycharm, ולהעתיק לתוכו את קבצי התוכנה כפי שעשינו בתדריך של האנדרואיד ולהריץ את התוכנה.

### המחשב הקרוב:

יבוצע במערכת הפעלה חלונות.

המחשב הקרוב ירוץ כתהליך עצמאי, ללא ממשק משתמש (אין צורך בקיוי).  
אנו נשתמש ב-Host (מערכת הפעלה חלונות) למטרה זאת.  
שינויים מהתוכנה המקורית:

- אין צורך ליבא את הספריות מקיוי, יש ליבא את redis.
  - בנוסף לספריות הקיימות מ-NXT יש גם ליבא את הספרייה next.sensor.
  - אפשר להשתמש במחלקה הקיימת (רצוי לשנות את השם מ-Mimshak, לשם משמעותי), ולשנות את המתודה start שתהפוך לבנאי של המחלקה. בבנאי נבצע את הדברים הבאים:
    - נקלוט ממפעיל התוכנה באמצעות raw\_input את שם הרובולגו.
    - נאתחל את הרובולגו ואת המנועים שלו כמו שעשינו בתוכנית המקורית.
    - ניצור חיבור לרדיס בצורה הבאה:
- ```
self.r=redis.StrictRedis('local host',6379,0, decode_responses=True,charset='utf-8')
```

כאשר הפרמטרים משמאל לימין הם: כתובת IP של שרת הרדיס אנו נשתמש בשרת רדיס מקומי (לכן ניתן לכתוב local host), פורט השרת, מספר מסד הנתונים (ניתן ליצור 4 מסדים שונים), רדיס שומר את הערכים כבייטים, תפקיד שני הפרמטרים האחרונים הוא להחליף את הבייטים למחרוזות.

- נכתוב את שם הרובוט ב-redis, ע"י הפקודה: `self.r.set('brickName',self.brickName)`
- ניצור מילון שמתרגם את ערכי החיישן לצבעים ע"פ הטבלה.
- ניצור חבר מחלקה שישמש כדגל שיפסיק את הלולאה האינסופית של השרת.
- יש ליצור מתודה במחלקה שתקרא את הצבע ישירות מהרובולגו, בדיקת הצבע נעשית ע"י הפקודה  
`next.sensor.Color20(self.brickName,next.sensor.PORT_1).get_sample()`

כאשר הפרמטר הראשון הינו שם הרובולגו כפי שכתוב בתוכנית המקורית.  
כתיבת הצבע ברדיס נעשית ע"י הפקודה:

```
self.r.set(self.brickName+':color',color)
```

עודכן ע"י גד הלוי 02/24 כל הזכויות שמורות לאוניברסיטת ת"א ולמחבר.  
אין להשתמש בתדריך זה ואז בחלקים ממנו ללא אישור המחבר.  
עמוד 3 מתוך 8

- כאשר הפרמטר הראשון הינו המפתח, והשני הערך.  
המפתח תמיד יתחיל בשם הרובולגו נקודותיים ושם התכונה המבוקשת. בכל פעם שניזין לרדיס את אותו המפתח הוא ידרוס את המפתח עם הערך הקודם.
- יש ליצור מתודה במחלקה שתקרא את הכיוון ברדיס קריאה מרדיס נעשית ע"י הפקודה  
`kivun=self.r.get(self.brickName+':kivun')`
- כאשר המפתח הינו שם הרובולגו, והתכונה היא `kivun`.  
 התוכנה תתרגם את הכיוון להוראה למנועים. לדוגמה: `self.yamin.run(-POWER)` נותן הוראה למנוע הימני לנסוע אחורה, כאשר `POWER` משמש כקבוע בתוכנה וערכו המומלץ בסביבות 60.
- הסרור אמור לרוץ נון-סטופ, לכן יש להוסיף את המתודה הבאה:

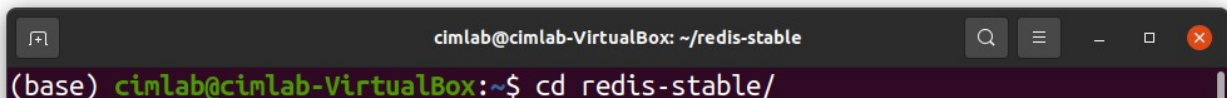
```
def main(self):
    while self.flag:
        self.findColor()
        time.sleep(DELAY)
        self.drive()
        time.sleep(DELAY)
```

שמפעילה לסירוגין את שתי המתודות, כאשר יש השהייה של 100 מילישניות בין הפעלה להפעלה. אנא נסו בשלב ראשון לכתוב את תוכנת המחשב הקרוב לבד, אם אתם נתקעים תוכלו להוריד את הקובץ `serverSulMid` ולכתוב את הפתרון בקובץ, אם עדיין תהיו תקועים הורידו את `serverSulEasy` וכתבו בו את הפתרון.

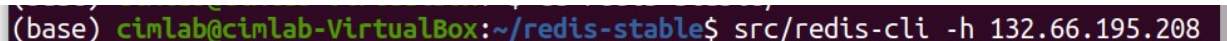
#### הפעלת לקוח רדיס מקומי:

במערכת הפעלה חלונות יש להכנס ל-`cmd` ובאמצעות הפקודה `ipconfig` יש למצוא את כתובת ה-`ip` של המחשב.

את תוכנת המחשב הקרוב שכתבנו נבדוק באמצעות תוכנת הלקוח של `Redis`.  
 להפעלת התכנה יש להיכנס לטרמינל (במחשב הרחוק **בלינקס**) ולהכנס לתיקייה של `Redis`:



להפעלת תוכנת הלקוח של רדיס יש לבצע את הפעולה הבאה כאשר כתובת ה-`IP` של מחשב ה-`Windows` שלכם תחליף את המספר שבתמונה:

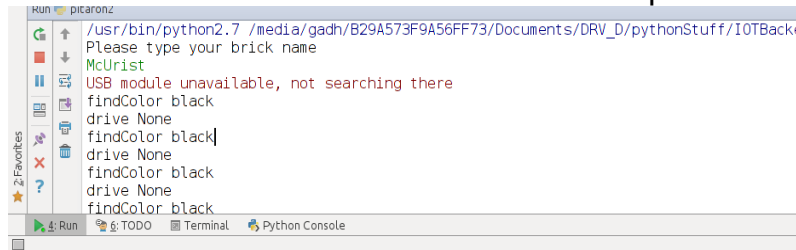


בטרמינל סמן הכתיבה מכיל את כתובת המחשב כולל הפורט. במסך המצ"ב ניתן לראות כיצד כותבים ערך וקוראים ערך מרדיס, בדוגמה למפתח `brickNme` כותבים את הערך `YLapid11`, באמצעות הפקודה `set`, ואח"כ קוראים את הערך `YLapid11` באמצעות הפקודה `get`.

```
132.66.195.202:6379> get brickName
"YLapid11"
132.66.195.202:6379>
```

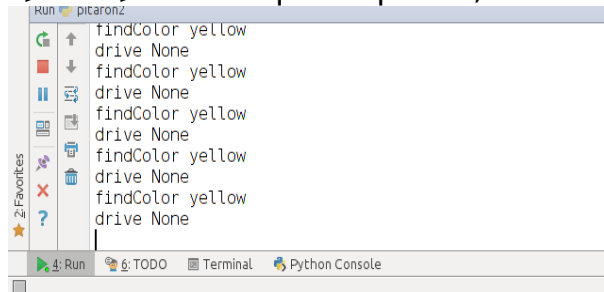
### בדיקת המחשב הקרוב:

במחשב הקרוב, יש להריץ את התוכנית, ולהקליד את שם הרובולגו במקום המתאים, לאחר מספר שניות תקבלו את המסך המצ"ב.



```
Run pitaronz
/usr/bin/python2.7 /media/gadh/B29A573F9A56FF73/Documents/DRV_D/pythonStuff/IOTBack
Please type your brick name
McUrist
USB module unavailable, not searching there
findColor black
drive None
findColor black
drive None
findColor black
drive None
findColor black
drive None
findColor black
drive None
```

יש לבדוק שחיישן הצבע עובד כראוי, הבדיקה במסך המצ"ב נעשתה עם הצבע הצהוב.



```
Run pitaronz
findColor yellow
drive None
findColor yellow
drive None
findColor yellow
drive None
findColor yellow
drive None
findColor yellow
drive None
findColor yellow
drive None
```

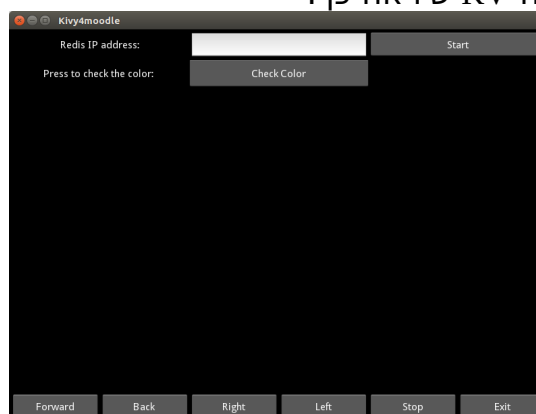
אנא בדוק שהמחשב הקרוב אכן יכול להזיז את הרובולגו, והצבע הנכון אכן נכתב במסד הנתונים. אם אינך יודע כיצד לבצע זאת אנא פתח במודל את הקובץ serverCheck.

### המחשב הרחוק:

יבוצע בלינוקס.

המחשב הרחוק לא מנהג ישירות את הרובולגו, אלא קורא ממסד הנתונים את הצבע, וכותב למסד הנתונים את הכיוון הרצוי, בהתאם לכפתור שהמשתמש לחץ.

- יש לשנות את ממשק ה-KV שיראה כך:



עודכן ע"י גד הלוי 02/24 כל הזכויות שמורות לאוניברסיטת ת"א ולמחבר.  
אין להשתמש בתדריך זה ואז בחלקים ממנו ללא אישור המחבר.  
עמוד 5 מתוך 8

שינויים מהתכנה שכתבתם בתדריך אנדרואיד:

- במתודה start נאתחל את רדיס בכתובת ה-IP של מחשב ה-Windows , וניקח את שם הרובולגו ממסד הנתונים.
- יש להוסיף שני כפתורים נוספים כמוראה בתמונה, הראשון Check Color יקרא את הצבע מרדיס ויכתוב אותו על המסך בתוך התוית (לייבל) מימין לכפתור, והשני הינו כפתור יציאה.
- יש לאתחל במתודה start את המצב התחילי ל-stop.
- לא לשכוח לעדכן את קובץ הפייטון ואת קובץ ה-KV. אנא נסו בשלב ראשון לכתוב את תוכנת המחשב הרחוק לבד, אם אתם נתקעים תוכלו להוריד את הקבצים kivyClientEasy ו-kvEasy ולכתוב את הפתרון בקובץ הפייטון.

#### בדיקת המחשב הרחוק:

- נסה לבדוק את המחשב הרחוק לבד.
- אם אינך יודע כיצד לבצע זאת אנא פתח במודל את הקובץ clientCheck.

#### בדיקת הקוד:

- כל קבוצה תריץ את המחשב הקרוב שבנתה במחשב שלה במערכת הפעלה חלונות 10 , ותריץ גם את המחשב הרחוק בקיוי בוירטואל בוקס בלינוקס.
- כל קבוצה באמצעות הטרמינל במערכת הפעלה Windows תברר מהו ה-IP של המחשב שלה, בטרמינל יש להקליד ipconfig ולראות מהו ה-ip.
- ה"משתמש" יקליד את ה-IP מהסעיף הראשון לתוך ממשק המשתמש בקיוי.
- יש לבדוק שאכן הצבעים הנכונים מוצגים למשתמש, והרובולגו נוסע לכיוון שהמשתמש בוחר.
- לאחר הבדיקה יש להמשיך בתדריך.

#### חלק שני:

- בחלק זה נפעיל את הרובולגו בעזרת ממשק אינטרנטי שלם, ללא השימוש בקיוי.
- אנו נשתמש בתכנות דינמי של שרת האינטרנט בחבילת התוכנה Dash.
- בעקרון מבנה היישום שלנו לא ישתנה, נשתמש בסרוור ובקליינט, כאשר הסרוור ישאר ללא שינוי מחלק הקודם, ורק הקליינט ישתנה. את התרגילים המודרכים שנמצאים במודל כדאי לבצע, ולהבין.

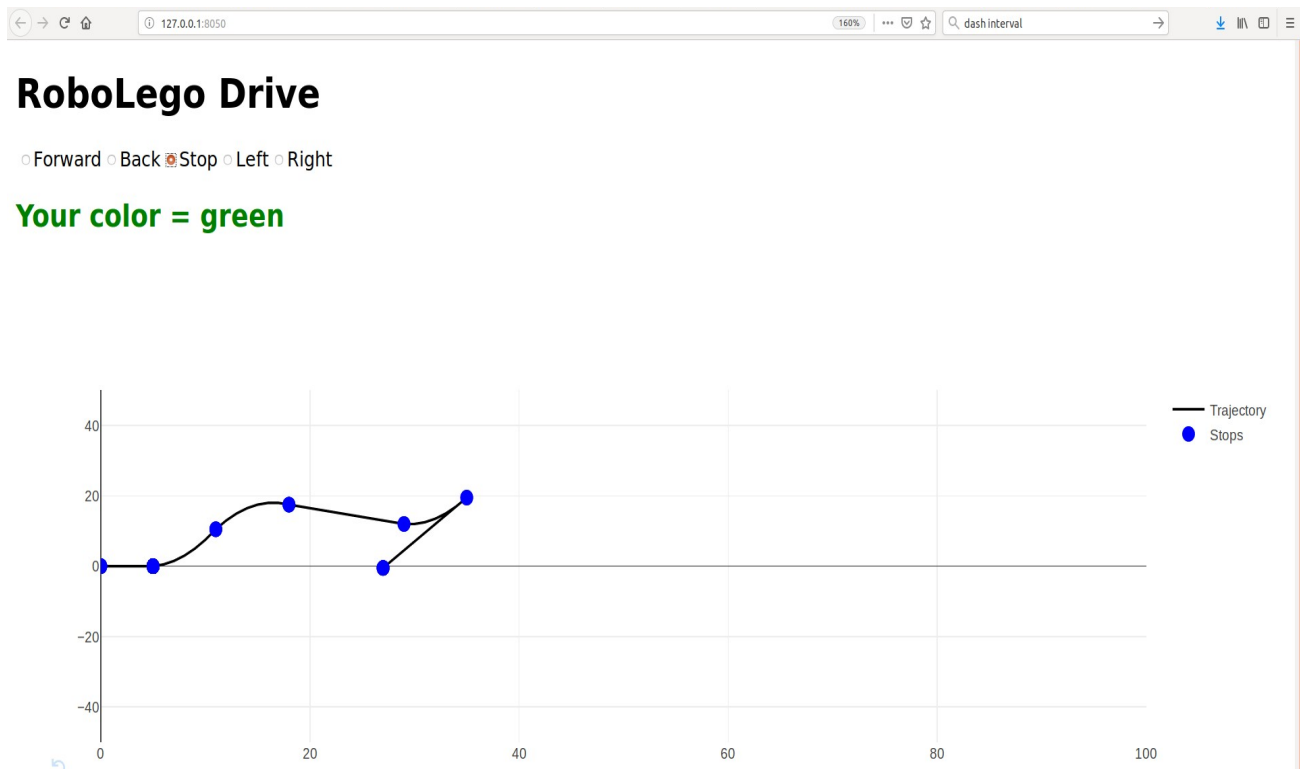
#### משימת סיום:

כתבו תכנית שבה:

1. המשתמש יבחר את הכיוון שאליו יסע הרובולגו בעזרת "לחצני רדיו" שנמצאים בספרייה dcc ונקראים dcc.RadioItems.
2. הצבע שקורא חיישן הצבע יופיע על המסך בצבע המתאים.
3. יש להציג בגרף את מסלול הרובולגו.

עודכן ע"י גד הלוי 02/24 כל הזכויות שמורות לאוניברסיטת ת"א ולמחבר.  
אין להשתמש בתדריך זה ואו בחלקים ממנו ללא אישור המחבר.

עמוד 6 מתוך 8



#### הסבר:

- הצבע הנכון יופיע על המסך ברגע שהחיישן יהיה מעליו.
- העצירות במסלול מצויינות ככדורים כחולים (Markers).
- לפני כל שינוי כיוון יש לבצע עצירה.
- הסבר המסלול: הרובולגו נסע ישר, פנה שמאלה, ימינה, ישר, שמאלה ואחורה.

#### הערות והמלצות:

- השרת שבנינו בסעיף הראשון נשאר ללא שינוי, בסך הכל בסעיף זה צריך לכתוב תכנית ש:
  - קולטת מהמשתמש את הכיוון הרצוי וכותבת אותו ברדיס.
  - קוראת את הצבע ממסד הנתונים ומציגה אותו למשתמש.

#### ורפייר:

- יש להסריט את האפליקציה הראשונה בקיווי עובדת בתור משימה 1.
- יש לשים את הקוד של משימה ראשונה בורפייר.
- יש להסריט את האפליקציה השניה ב-Dash עובדת בתור משימה 2.
- יש לשים את הקוד של משימה שניה בורפייר.

עודכן ע"י גד הלוי 02/24 כל הזכויות שמורות לאוניברסיטת ת"א ולמחבר.  
 אין להשתמש בתדריך זה ואז בחלקים ממנו ללא אישור המחבר.  
 עמוד 7 מתוך 8

# בהצלחה!

עודכן ע"י גד הלוי 02/24 כל הזכויות שמורות לאוניברסיטת ת"א ולמחבר.  
אין להשתמש בתדריך זה ואז בחלקים ממנו ללא אישור המחבר.  
עמוד 8 מתוך 8