**מדריך לפרויקט והוראות הפעלה:**

כל הקוד של הפרויקט נמצא כאן - https://github.com/potashkeren/BPBlockly

1. **הסבר על התיקיות והקבצים:**

תחת הפרויקט ישנן 2 תיקיות עיקריות:

1. blockly-editor
2. src

* **תיקיית blockly-editor**

תיקייה זו אחראית על כל הקוד של Blockly: הבלוקים השונים, דוגמאות בקבצי xml ודוגמאות בשפת JavaScript. פירוט התיקיות השונות:

* + blockly-files – מחולק לשני סוגים של קבצים
    - קבצי blockly – כל קבצי הsource-code של פרוייקט Blockly (הסבר מלא על הקבצים ניתן למצוא כאן - <https://developers.google.com/blockly/guides/modify/web/building> <https://developers.google.com/blockly/guides/app-integration/running-javascript>)
      * acorn\_interpreter.js – תמיכה בJS-Interpreter
      * blockly-style.css – קובץ הcss עבור הבלוקים השונים.
      * blockly\_compressed.js – כולל את הקוד להפעלת הבלוקים הקיימים שנמצא תחת תיקיית "core/".
      * blocks\_compressed.js – כולל את כל התוכן של הבלוקים הקיימים תחת תיקיית "blocks/".
      * en.js – תמיכה בBlockly בשפה האנגלית.
      * FileSaver.js – קובץ עזר לשמירה של קבצים עם בלוקים של Blockly
      * Index.html – הקובץ שאחראי על תצוגת הGUI למשתמש. בקובץ זה נמצאים כל קבצי ה-script המתוארים כאן לצורך בניית והרצת פרויקט עם Blocky. מוגדרים כאן הכפתורים לשמירה וטעינה של הדוגמאות והכפתור להרצת התוכנית. לאחר יצירת הבלוקים בקובץ "our\_blocks.js", בשביל להציג את הבלוק בGUI, צריך להוסיף את הבלוק גם לקובץ זה תחת תגית ה-"toolbox" האחראית על תפריט הצד שבו המשתמש יכול לבחור את הבלוקים שאיתם יעבוד (הסבר מלא תחת סעיף – יצירת הבלוקים)
      * listeners.js – כאן נמצא הקוד של ה-listeners האחראים על שמירת קבצי התוכנית, טעינתם והרצתם באמצעות הכפתורים השונים שבממשק המשתמש.
      * our\_blocks.js – בקובץ זה ניתן ליצור את הבלוקים המותאמים אישית לפי המדריך הבא - <https://developers.google.com/blockly/guides/create-custom-blocks/overview>. מדריך זה הינו מדריך ה- Blockly למפתחים כולל הסבר וממשק API ליצירת בלוקים מותאמות אישית . הסבר מלא על אופן יצירת הבלוקים ושילובם בממשק המשתמש – תחת סעיף "יצירת הבלוקים".
    - קבצי דוגמאות –

כל הדוגמאות השונות שיוצרו באמצעות הבלוקים של Blockly נשמרות כקבצי xml. לאחר הרצת התוכנית (הסבר מלא תחת סעיף – "הרצת התוכנית"), ניתן ללחוץ בממשק המשתמש על כפתור "Load Project", לבחור את אחת הדוגמאות מהתיקייה ולראות את הבלוקים הרלוונטיים לאותה דוגמא. לאחר סיום העבודה על הדוגמא, ניתן ללחוץ על כפתור "Save Project As.." ולשמור את הדוגמא לצורך עבודה עתידית.

* + examples
    - תחת תיקייה זו שמורות הדוגמאות השונות בשפת JavaScript.
  + TicTacToe
    - תחת תיקייה זו נמצאים קבצי הhtml וה-css עבור הGUI של משחק הTicTacToe
  + user
    - דוגמאות נוספות בשפת JaveScript.
* **תיקיית src**

תיקייה זו אחראית על הרצת הדוגמאות שיצרנו, הרצת ה-BP Server, ועבודה עם בסיס הנתונים.

* + תחת תיקיית src\main\java\il\ac\bgu\bp\cotextualBlockly נמצאים הקבצים והתיקיות הבאים:
    - תיקיית context – בה מוגדרים כל האובייקטים הקונטקסטואליים עבור הרצת דוגמאות בשפת COBP. בנוסף קיימים הקבצים – "TimeInjector" ו- "SimulatedTimeInjector" אשר משמשים להרצת דוגמאות מבוססות זמן (דוגמת ה-Lab management system)
    - BPServer.java – הקובץ המשמש להרצת התוכנית, כדי לעלות את ממשק המשתמש, עלינו להריץ את קובץ זה, כך נוכל להיכנס לקישור המופיע בconsole- (<http://localhost:8080/blockly-editor/blockly-files/index.html>) ולעבוד עם הבלוקים שיצרנו.
    - ContextInstance.java – קובץ זה אחראי על הרצת תכניות BP, עבודה עם External events ועבודה עם דוגמאות מבוססות זמן באמצעות ה-timeInjector.
    - DebugEndPoint.java – הוספת debug endpoint ל-server container.
    - EventQueue.java- אחראי על הטיפול ב-events השונים והדפסתם לconsole.
    - RunServlet.java – קובץ המשמש להרצת HttpServlet.
    - TestRun – באמצעותו ניתן להריץ את הטסטים השונים שנכתבו עבור התוכנית הרצויה ע"י הוספת הקבצים הרלוונטים לאובייקט "ContextInstance".
  + תיקיית src\main\resources
    - תיקיית META-INF – תחת תיקייה זו נמצא קובץ "persistence.xml" האחראי על הגדרת העבודה עם הבסיס נתונים באמצעות ספריית Hibernate המספקת תשית למיפוי בין אובייקטים מונחי עצמים לבסיס נתונים רלציוני
    - תיקיית TTT – תחת תיקייה זו שמורים קבצי קוד ה-JavaScript של ה-case study של משחק ה-TicTacToe (מימוש ללא קונטקסט, המימוש הקונטקסטואלי למשחק ויצירת ה-DB).
    - שאר הקבצים בתיקייה משמשים הינם הקבצים הרלוונטים לצורך הרצת ה-case-study של דוגמת ה-Lab management system.

1. **יצירת הבלוקים**

המדריך המלא ליצירת הבלוקים המותאמים אישית - <https://developers.google.com/blockly/guides/create-custom-blocks/overview>

יצירת הבלוקים השונים מורכבת משלושה שלבים עיקריים:

1. **Define Blocks** –

הגדרת הבלוק נעשית תחת קובץ "our\_blocks.js" ומתארת כיצד הבלוק נראה ומתנהג, כולל הטקסט, הצבע, הצורה, ובלוקים נוספים שאליהם הבלוק יכול להתחבר.

את אובייקט הבלוק ניתן להגדיר תחת " Blockly.defineBlocksWithJsonArray" (בפרוייקט שלנו כל הבלוקים מוגדרים כאובייקטים בפורמט JSON, אך ניתן להגדירם גם בפורמט JavaScript).

כל המאפיינים השונים של הבלוק מפורטים בקישור הנ"ל , כאשר ה-"type" מגדיר את השם של הבלוק: <https://developers.google.com/blockly/guides/create-custom-blocks/define-blocks>

1. **Generating Code** –

המרת הבלוק לקוד JavaScript הרצוי באמצעות "Blockly.JavaScript[block\_name]" תחת קובץ "our\_blocks.js". את המשתנה "block\_name" נחליף בשם הבלוק כפי שהגדרנו אותו תחת המאפיין "type" בסעיף הקודם. כל פונקציה שניצור, תקבל את הבלוק והמאפיינים שלו ונוכל לגשת אל המאפיינים ולתרגם אותם לקוד ה-JavaScript התואם את התנהגות הבלוק.

ניתן להמיר את הבלוקים גם לשפות - Python, PHP, Lua, Dart.

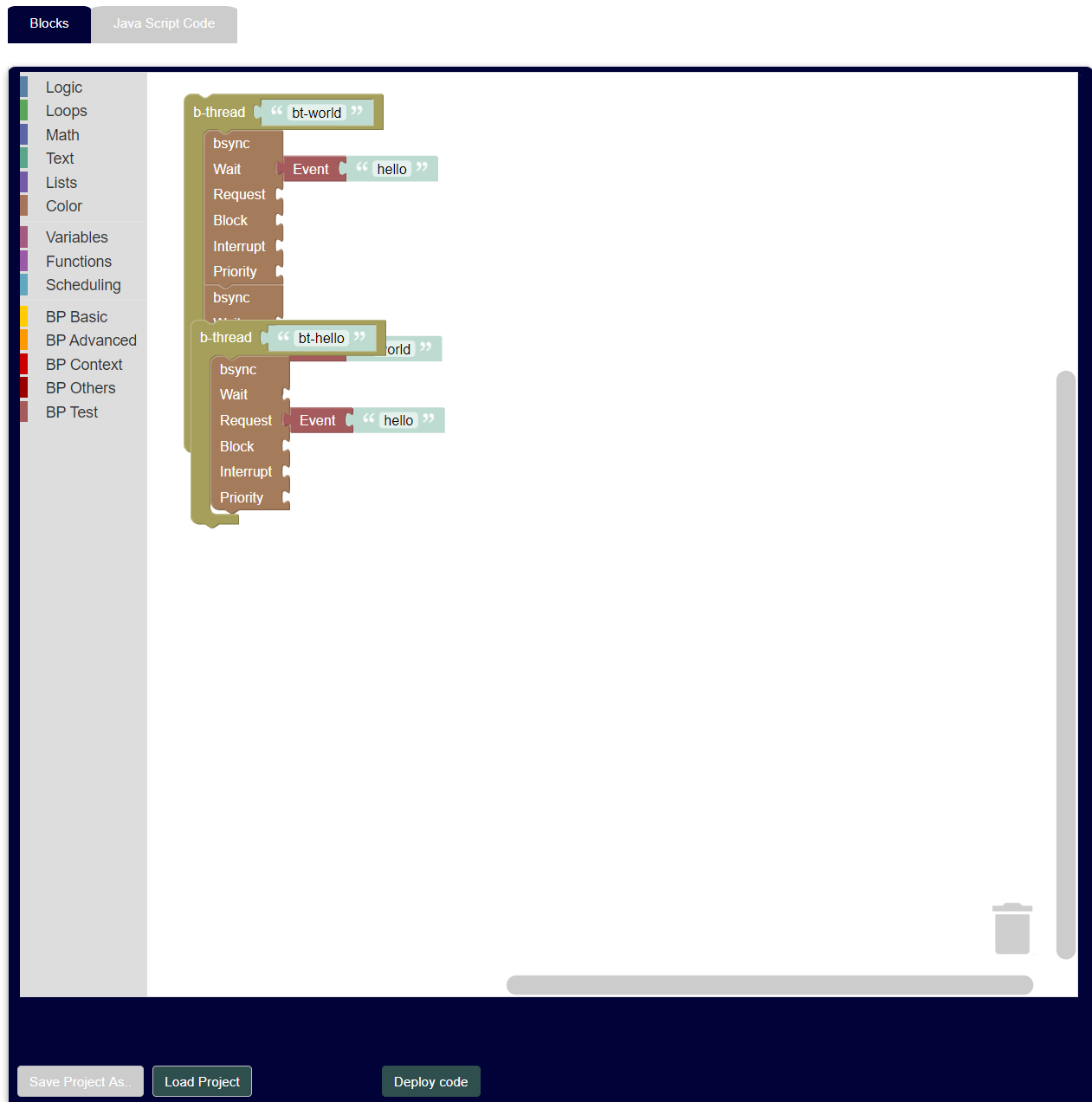
למדריך המלא - <https://developers.google.com/blockly/guides/create-custom-blocks/generating-code>

1. **Add Blocks to Menu** –

לאחר שהגדרנו את מאפייני האובייקט של הבלוק והגדרנו את הקוד JavaScript שיתורגם מאותו בלוק, עלינו להוסיף את הבלוק לתפריט הצד בממשק המשתמש שבו נמצאים כל הבלוקים.

תחת קובץ blockly-editor/blockly-files/index.html בתוך תגית הxml- עם ה-id="toolbox", נמצאות כל הקטגוריות של הבלוקים השונים. ניתן להוסיף קטגוריות חדשות לפי רצון המשתמש, וניתן להוסיף את הבלוקים שיצרנו תחת הקטגוריה הרלוונטית. הוספת הבלוק נעשית באמצעות התגית <block> כאשר השדה "type" מייצג את שם הבלוק. לכל הגדרה של בלוק ניתן להוסיף שדות דיפולטיביים שיוצגו למשתמש.

1. **הרצת התוכנית**
2. כדי להרים את התוכנית עלינו להריץ את קובץ ה-BPServer.java.
3. לאחר מכן ניתן להיכנס לקישור המופיע ב-console - <http://localhost:8080/blockly-editor/blockly-files/index.html> אשר פותח את סביבת העבודה למשתמש, ולהתחיל ליצור את התוכנית באמצעות הבלוקים של Blockly.
4. סביבת העבודה נראית כך:



ומורכבת משני חלונות:

1. Blocks – כאן ניתן לעבוד עם הבלוק השונים כדי ליצור את התוכנית הרצויה, ניתן לטעון תוכניות ישנות באמצעות הכפתור "Load Project" ולשמור תוכניות שעבדנו עליהם כעת באמצעות הכפתור "Save Project As".
2. Jave Script Code – בחלון זה מופיע כל הקוד שמתורגם מהבלוקים שהשתמשנו בהם.
3. לאחר שסיימנו ליצור את התוכנית הרצויה, נוכל להעתיק את כל הקוד שמופיע בחלון " Jave Script Code" ולשמור אותו כקובץ JavaScript בפרויקט.
4. את הקובץ ששמרנו בפרויקט ניתן כעת לטעון אותו באמצעות כתיבת שם הקובץ בתוך האובייקט " ContextInstance" תחת הקובץ "BPServer" בשורה 53.
5. לאחר מכן נוכל להריץ את כל הפרויקט בשנית (run BPServer) וללחוץ על כפתור Deploy code אשר יריץ את התוכנית שיצרנו.