פרויקט תכנות מתקדם תשפ"ד – תרגיל 6

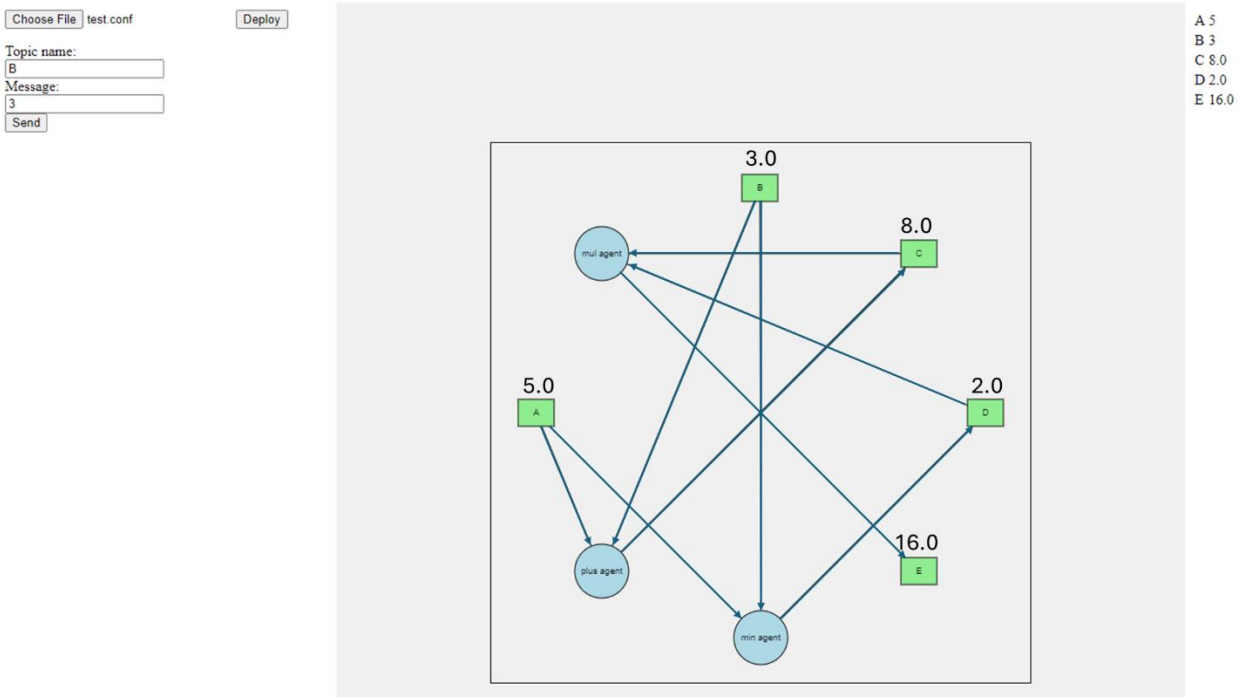
רקע

לאחר תרגילים 1-5 הגיעה העת לכתוב את שכבת ה view. שכבת ה view תורכב מקובצי HTML, חלקם סטטיים, חלקם מג'ונרטים ע"פ הבקשות לשרת. כזכור, כל "מכונה" מריצה שרת לוקאלי שאליו המפעיל יכול להתחבר מרחוק באמצעות דפדפן, לראות את התצוגה שנרכב בתרגיל זה, ודרכה לשלוח קונפיגורציה שתרוץ על המכונה. בנוסף נאפשר לו לבדוק אותה ע"י שליחת הודעה ל Topic ספציפי ולראות את התוצאה בגרף.

תרגיל זה מגדיר את דרישות הסף המינימליות ביותר ומשאיר לכם הרבה מקום לפתח עוד פיצ'רים אפשריים וכמובן לשלוט בנראות של התצוגה. זכרו, זהו חלון הראווה שדרכו מעסיקים פוטנציאליים ישפטו במהירות את הפרויקט ויקבלו החלטה האם להעמיק בו עוד.

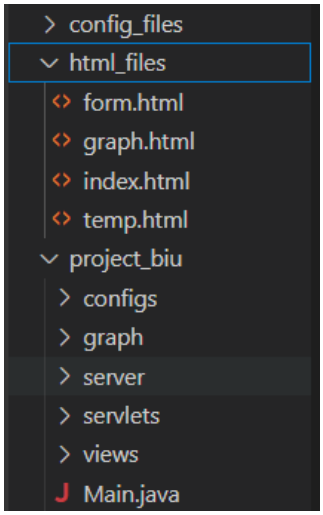
התצוגה הכללית תחולק ל 3 חלקים:

- בצד שמאל שני טפסים קטנים,
  - האחד מאפשר טעינה של קונפיגורציה
  - השני מאפשר שליחה של הודעה ל topic
- בחלל המרכזי יוצג הגרף החישובי שבחרנו לטעון
- בצד הימני טבלת הערכים הנוכחיים של כל Topic



קובצי HTML

צרו תיקייה מחוץ לתיקיית קובצי המקור בשם html_files וצרו שם את הקבצים הבאים. מבנה הפרויקט הסופי צריך להיראות אצלכם כך:



בקובץ index.html עליכם להגדיר 3 inner frames (תגית iframe) עבור

- תצוגת הקובץ from.html מצד שמאל אשר יכיל את שני הטפסים
- תצוגת הגרף במרכז אשר כברירת המחדל מציגה את temp.html
- תצוגת טבלת הערכים אשר כברירת המחדל גם מציגה את temp.html

:from.html

בקובץ זה עליכם ליצור (לפחות) שני טפסים:

- טופס לבחירה והעלאה של קובץ קונפיגורציה.
  - הפעולה צריכה להיות מופנית ל http://localhost:8080/upload
  - בשיטת post
  - על כפתור השליחה צריך להיות כתוב deploy
  - הפלט יוצג בחלל המרכזי
- טופס לשליחה של הודעה ל topic.
  - הפעולה צריכה להיות מופנית ל http://localhost:8080/publish
  - בשיטת get
  - הפלט יוצג בטבלה שבצד שמאל

רצוי להוסיף טפסים נוספים להפעלת המכונה מרחוק.

קובץ temp.html הינו קובץ html ריק שנמצא שם רק כדי למלא את החלל עד אשר תתבצע פעולה.

לכל אלו אתם בהחלט מוזמנים להשתמש ב chat gpt או דומיו כדי ליצור את קבצים אלו ולחסוך בזמן פיתוח.

קובץ graph.html

צריך להציג תצוגה גרפית של גרף מכוון המייצג את הגרף החישובי שנטען. את הקובץ הזה אתם למעשה מייצרים באמצעות הקוד בשרת לאחר שנבחר קובץ הקונפוגורציה ע"י המשתמש (מאוחר יותר נראה את התהליך).

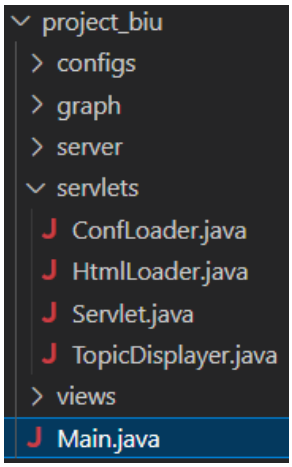
כאן יש לכם מלאכת **למידה עצמאית**. אתם יכולים להשתמש ב canvas של html 5 או ב java script או בספריות קוד-פתוח מוכנות לציורים של גרפים. מה שתרצו. הדרישה המינימלית היא תצוגה בה ה topics הם מלבניים ואילו הסוכנים עגולים. השמות שלהם כתובים בפנים, יש להם גם צבעים שונים, והחיצים ביניהם מכוונים ע"פ הכיוון בגרף החישובי.

בדוגמה לעיל אנו רואים את החישוב (A+B)*(A-B).

מעבר לזה אתם מוזמנים "להשתולל" בתצוגה. תוכלו למשל להשתמש בספרייה שמאפשרת הזזה דינאמית של הקודקודים בגרף, לכתוב את הערכים הנוכחיים על ה topics, לכתוב את החישוב כביטוי מתמטי ועוד.

מומלץ להתחיל מקובץ סטטי רגיל שכתוב ידנית, וכשתהיו מרוצים ממנו, להטמיע אותו בצורה גנרית בקוד שמייצר אותו (כפי שנראה בהמשך).

Servlets



כזכור, כדי לתת מענה לפקודות השונות במערכת יצרנו את הממשק Servlet. כעת הגיעה השעה לממש כמה מימושים שונים לממשק זה אשר אותם יטען השרת כמענה לפקודות בטפסים לעיל.

בואו נתחיל מהסוף, ה Main של הפרויקט צריך להיראות למעשה כך:

- יוצר מופע של HttpServer
- מוסיף 3 Servlets לשרת
  - אחד עבור תצוגת הטבלה של ה Topics
  - אחד עבור טעינה של קונפיגורציה
  - ואחד עבור טעניה כללית של קובץ html
- מפעיל את השרת ברקע
- ממתין לקלט שלאחריו סוגר את השרת.

```java
public class Main {
    public static void main(String[] args) throws Exception{

        HTTPServer server=new MyHTTPServer(8080,5);

        server.addServlet("GET", "/publish", new TopicDisplayer());
        server.addServlet("POST", "/upload", new ConfLoader());
        server.addServlet("GET", "/app/", new HtmlLoader("html_files"));

        server.start();
        System.in.read();
        server.close();
        System.out.println("done");

    }
}
```

לפיכך עליכם לממש:

- את TopicDisplayer כ Servlet שנותן מענה לטופס השני ב form.html לפקודה /publish
  - יחלץ את נתוני ה Topic וה Message מפקודת ה http
  - יפרסם את ההודעה ל topicהמתאים באמצעות ה TopicManager (הסינגלטוני כזכור)
  - יחזיר תשובת http תקינה שכוללת תוכן של html ובו טבלה עם 2 עמודות:
    - שם ה topic
    - הערך האחרון שהיה ב topic זה.
- את ConfLoader כ Servlet שנותן מענה לטופס הראשון ב form.html לפקודה /upload
  - יחלץ את שם הקובץ ואת תוכנו, וישמור את התוכן בצד השרת
  - יטען GenericConfig בהתאם לתוכן הקובץ, ויצור מופע של Graph מהקונפיגוריציה.
  - יחזיר תשובת http תקינה שמכילה את ה HTML של התצוגה הגרפית של הגרף החישובי.
    - יש להאציל את הסמכות הזו למחלקה אחרת כפי שנראה בהמשך.
- את HtmlLoader כ Servlet שנותן מענה לכל כתובת שמתחילה ב /app/.
  - יחלץ מהכתובת (URISegments, זוכרים?) את שם קובץ ה html שהלקוח ביקש

- יטען את הקובץ אם הוא קיים
  - אם הוא לא קיים, אז יש להחזיר תשובת HTML שמעידה שהקובץ לא נמצא.
- יחזיר תשובת http תקינה עם התוכן ה html שנטען מהקובץ
- **הערה:** שימו לב שמחלקה זו לא אמורה להכיר את התיקייה בה נמצאים קובצי ה html כמשהו קבוע בקוד (hard coded) אלא זהו פרמטר של הבנאי שלה, שכן הגדרה זו עשויה להשתנות מפרויקט לפרויקט. בדוגמת ה main לעיל הבנאי קיבל את התיקייה html_files שבה נמצאים קובצי ה html.

<u>שכבת ה view</u>

עד כה עסקנו בשכבת ה controller - אוסף של servlets שנטענים ע"פ בקשת הלקוחות ומבצעים פעולות שונות במערכת. קובצי ה html שיצרנו עד כה היו משאב סטטי של המערכת. ניתן במובן מסוים גם לשייך אותם ל view אך כאשר אנו מדברים על שכבת ה view אנו מתכוונים לקוד שמייצר לנו view ע"פ בקשה.

אנו נדגים זאת בפרויקט זה ע"י מחלקה אחת בשם HtmlGraphWriter שתיצרו ב package בשם views.

במחלקה זו עליכם לממש מתודה סטטית אחת בשם getGraphHTML אשר בהינתן מופע של Graph היא תחזיר רשימה של מחרוזות. הרשימה הזו מכילה תוכן HTML-י שמציג את הגרף החישובי בצורה גרפית נאה לעין. למעשה מדובר בקוד Java שמייצר קוד HTML ובתוכו אולי אלמנטים נוספים של Javascript בהתאם לצורך ובתוכו הוא שוזר את הערכים הרלוונטיים שנטענו מאותו מופע של Graph.

כך תוכלו להשתמש בפונקציה זו במחלקה ConfLoader – לאחר שיצרתם מופע של Graph תוכלו להחזיר HTML שמצייר אותו.

המלצה:

לא נרצה לייצר את כל הקוד ה HTML-י בצורה שהיא hard coded בתוך הפונקציה ה Java-אית הזו, שכן אם מחר נרצה לשנות דבר מה זה יהיה לשנות קוד במערכת (לקמפל, לבדוק, להפיץ וכו').

הרבה יותר נכון לכתוב קוד שטוען משאב סטטי (למשל את graph.html), ואז משנה בפנים נתונים שונים בהתאם לצרכים שלנו (מופע ה Graph) ואז מחזיר את התשובה המעודכנת.

כך בעת הצורך את התשתית וכל האלמנטים העיצוביים ניתן לשנות באותו משאב סטטי חיצוני (graph.html) ולא בקוד של המערכת, וזאת ללא צורך לבנות את הפרויקט מחדש.

הגשה:

הפעם את הקוד אתם לא מגישים למערכת הבדיקות, אלא לתיבת הגשה ייעודית במודול.

לתיבת ההגשה עליכם להגיש את הקובץ link.txt ובו

- לינק ל GIT שלכם שבו נמצא הפרויקט.
- ת.ז ומייל של שני המגישים

הפעם **<u>ההגשה היא זוגית</u>** (עדיין ניתן להגיש כבודדים למי שרוצה, אבל עדיף לתרגל את העולם האמתי בו עליכם לחבר קטעי קוד של אנשים שונים לכדי פרויקט יציב אחד). אך לא יותר מ 2.

בנוסף, כדי לדמות עוד קצת את העולם האמתי, הגשה זו כוללת מעט מהתוצרים שמלווים פרויקט אמיתי. תוצרים אלה מהווים חלק מהציון ומפורטים בעמוד הבא.

תוצרים ב GIT:

- קוד המקור שלכם כפי שמחולק ל packages המוגדרים לפרויקט
  - הקוד צריך להכיל מספיק הערות היכן שמתבקש
  - קטעים מהקוד יבדקו ידנית עבור אלמנטים שונים של SOLID
- ב readme הראשי עליכם לתאר את הפרויקט כמיטב יכולתכם ולכל הפחות לכל תיאור של:
  - רקע
  - התקנה
  - פקודות להרצה
  - **סרטון דמו:**
    - לא יותר מ 5 דק'
    - שקף פתיחה לפרויקט שכולל את פרטי הקורס ופרטי המגישים
    - שקף לתיאור סיפור הרקע
    - שקף לתיאור ה design של הפרויקט
    - דמו חי של הפיצ'רים העיקריים לפרויקט בדגש על תרגלי 6.
    - תדגישו דברים שעשיתם מעבר לדרישות המינימליות
    - אל תפחדו להראות גרפים מסובכים, שאולי גם מבצעים עיבודים לאו דווקא מתמטיים.
    - שקף סיכום של הדברים שלמדתם מהקורס ותיקחו איתכם הלאה
- תיעוד – Javadoc
  - בפרויקט כתבנו חיקוי של שרת HTTP פשוט שמפתחים אחרים יכולים לעשות לו reuse כספריית קוד. עליכם לחלץ Javadoc מלא שמפרט על השימוש ב API שיצרתם.
  - חישבו על מפתח שלא מכיר את הקוד הזה כלל ורוצה לעשות בו שימוש נכון.
    - למעשה כדאי לכם לקבל משוב מאחד כזה, תראו אם הוא מצליח לבדו להרים שרת שעושה דבר מה.
  - אז לא לשכוח אף תלות, וכדאי מאד לתת דוגמאות למפתחים.
- כל דבר אחר שמציג את היכולות שרכשתם בקורס עבור המעסיק הפוטנציאלי הבא שלכם.

הבדיקה של תרגיל 6 היא למעשה בדיקה ידנית של כל הפרויקט ותוצריו דרך ה GIT.

בהצלחה!