

תרגיל 12 "תוכנה 1": Servlets

מבוא

כאשר אתם גולשים באתר, הדפדפן שלכם מעביר בקשות לשרת מרוחק ומקבל ממנו תשובות. בדרך כלל הדפדפן שולח בקשה בכל פעם שאתם מקליקים על קישורית וכל פעם שאתם מקליקים על כפתור ששולח נתונים חזרה לשרת. (יש גם אתרים שבהם הדפדפן שולח בקשות לעיתים יותר קרובות, אבל לא נדון בהם בתרגיל). התשובה של השרת היא בדרך כלל דף אינטרנט חדש (גם כאן יש יוצאים מן הכלל). מה קורה בצד של השרת כאשר הוא מקבל בקשה כזו? לפעמים ההתנהגות של השרת פשוטה מאוד, והוא פשוט שולח לדפדפן קובץ ששמור בשרת. בעזרת התנהגות כזו צפיתם באתר האינטרנט של הקורס (שהוא פשוט קובץ html שאנחנו עורכים בעזרת מעבד תמלילים) ובעזרת התנהגות כזו הורדתם את התרגיל הזה.

אבל לפעמים ההתנהגות של השרת צריכה להיות מורכבת יותר ודורשת הרצת תוכנה שתחשב את התשובה שתוצג לכם. בתרגיל הזה נפתח תוכנה ששרת מפעיל כדי לספק תשובה לדפדפן.

יש כמה מנגנונים שבעזרתם שרת יכול להפעיל תוכנה שאמורה לספק תשובה לבקשה של דפדפן. המנגנון שנשתמש בו בתרגיל הזה נקרא Servlets (שרתונים).

Servlets

Servlet היא מחלקה שהקוד שלה נמצא על השרת. מחלקות כאלה תמיד מרחיבות מחלקה קיימת שנקראת `javax.servlet.http.HttpServlet`. (למען הדייק הן יכולות להרחיב עוד מחלקה אחת, אבל זה לא רלוונטי לתרגיל הזה.) דפדפן יכול להפעיל שירותים של המחלקה מרוחק, על ידי שליחת בקשה לשרת, בקשה שבה המשאב הנדרש הוא שם המחלקה. הדפדפן יכול גם להעביר פרמטרים לשירותים של המחלקה.

השירותים העיקריים של servlet הם `doGet`, `doPost` ו-`service`. השירות הראשון, `service`, הוא הכללי ביותר. הוא מופעל בכל פעם שהשרת מקבל בקשה מדפדפן להפעיל את המחלקה. מועברים לו שני ארגומנטים, האחד מטיפוס `HttpServletRequest` והשני מטיפוס `HttpServletResponse` (שניהם בחבילה `javax.servlet.http`, כמו שאר הטיפוסים שנשתמש בהם בתרגיל ושקשורים ל-servlets). שני הטיפוסים הללו הם ממשקים (interfaces).

הארגומנט הראשון, מטיפוס `HttpServletRequest`, מתאר את הבקשה ששלח הדפדפן. שני שירותים חשובים שטיפוס הזה מספק הם `getMethod()` שמחזירה מחרוזת שמתארת את סוג הבקשה ששלח הדפדפן (`GET`, `POST` או `PUT`). בפרוטוקול התקשורת HTTP, שהדפדפן והשרת מתקשרים בעזרתו, יש מספר סוגי בקשות. הנפוצות ביותר הן `GET` ו-`POST`. הראשונה משמשת בעיקר להעברת בקשות שאין להן הרבה פרמטרים פרט לכתובת, ה-URL. השנייה משמשת בעיקר להעברת בקשות שנושאות הרבה מידע (למשל קובץ ששולחים לשרת) או הרבה שדות מידע, כמו בטופס מורכב. הערכים של הבקשה יופיעו כזוגות של `key-value` לאחר הכתובת של הדף שאתם מבקשים.

השירות החשוב השני של `HttpServletRequest` הוא `getParameter(String s)`, בעזרתו נקבל את הערך של אחד מהפרמטרים שהועבר בבקשה. השירות מקבל את שם הפרמטר שהגיע כחלק מבקשת הדפדפן ומחזיר את ערכו. פרמטר כזה יכול להיות ערך שהמשתמש מילא בשדה קלט,

למשל. בנוסף, קיים השירות `getQueryString()` המחזיר את כל המחרוזת שלאחר כתובת הדף ללא עיבוד מקדים.

לדוגמה: אם בקשנו את הדף

`http://lab023.cs.tau.ac.il:40566/AddressBookServlet?name=Dana%20Levi`

`http://lab023.cs.tau.ac.il:40566/AddressBookServlet` היא כתובת הדף, `name=Dana%20Levi` היא מחרוזת השאילתה. מחרוזת זו מורכבת מפרמטר יחיד בשם `name` שערכו הוא "Dana Levi".

`HttpServletResponse`, שמייצג את התשובה שתשלח לדפדפן, גם יש שני שירותים חשובים. אחד הוא הפקודה `setContentType(String s)`, שדרכה מודיעים לדפדפן את סוג הקובץ שנשלח כתשובה. הערך הנפוץ ביותר לסוג הקובץ הוא "text/html", קובץ `html`, שבו אנחנו נשתמש. השירות החשוב השני הוא השאילתה `getWriter()`, שמחזירה עצם מטיפוס `PrintWriter`, אותו טיפוס של `System.out` שהשתמשנו בו בתרגילים קודמים. הדפסה על העצם הזה באמצעות השירות `println` ודומיו בונה את התשובה (דף ה `html`) שתשלח לדפדפן. בסיום השימוש בעצם הזה צריך לקרוא לשירות `close()` שלו.

השירותים `doGet` ו-`doPost` פועלים בצורה דומה, אלא שכל אחד מהם נקרא רק עבור בקשות מתאימות של הדפדפן. יש ל-`HttpServlet` שירותים דומים עבור כל סוגי בקשות ה-HTTP.

הכנת שרת

כדי לפתח את התרגיל, דרוש שרת HTTP שמסוגל להריץ `servlets`. אין צורך להבין מהו שרת HTTP או כיצד הוא עובד, פשוט עכבו אחר ההוראות המפורטות למטה כדי ליצור את סביבת העבודה.

בחלק ממחשבי הלינוקס של בית הספר מותקן שרת HTTP, ששמו `Tomcat`, בו נשתמש על מנת להריץ את התרגיל. אתם יכולים גם להתקין את השרת הזה בבית (אפשר להוריד אותו חינם), אבל קל יותר להשתמש בשרת המותקן בבית הספר, ורק בו סגל הקורס יכול לתמוך.

ראשית, פיתחו חלון `shell` בלינוקס, בספרית הבית שלכם, כעת עליכם להתחבר לאחד מהשרתים המריצים את `tomcat`. יש שניים עשר שרתים כאלו ושמותיהם `ibmop-01` עד `ibmop-12`. כדי להתחבר לאחד מהשרתים הנ"ל עליכם להריץ את הפקודה:

```
ssh ibmop-??
```

כאשר במקום `??` הכניסו את מספרו של אחד השרתים (לדוגמה `ibmop-01`). כעת תדרשו להזדהות שוב.

לאחר שהתחברתם לאחד משרתי `ibmop` הריצו את הפקודה

```
create-my-tomcat ~/tomcat-sw1 40XXX 50XXX
```

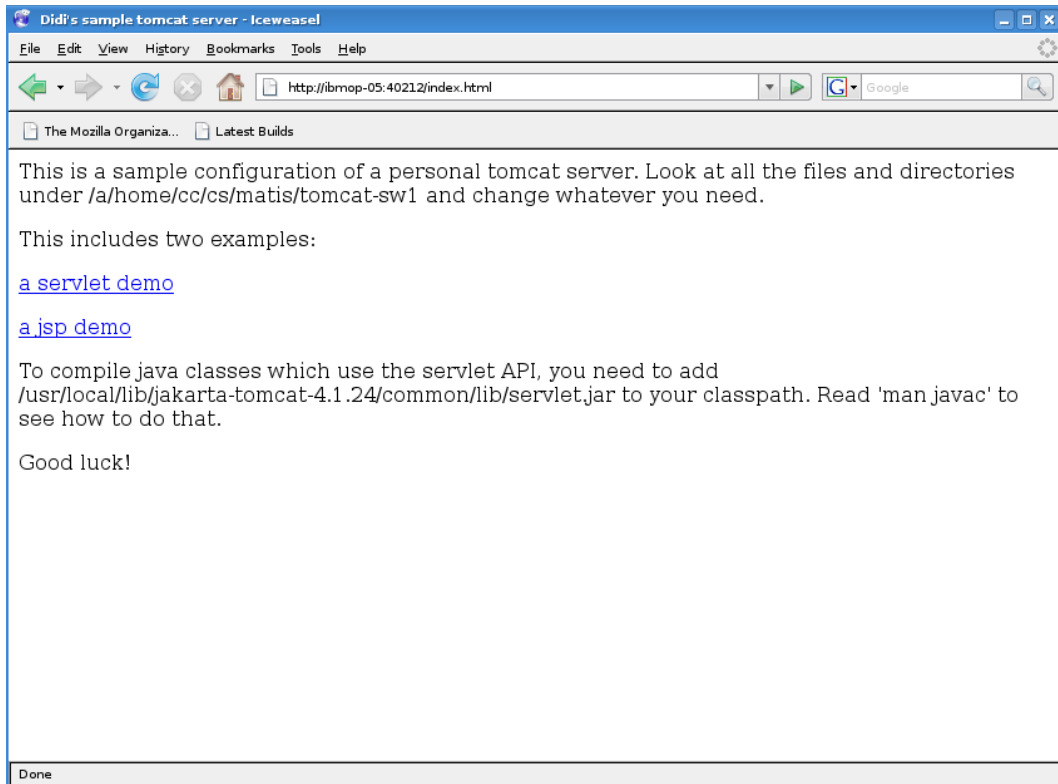
`XXX` הן שלוש ספרות כלשהן. הפקודה הזו פשוט יוצרת ספרייה בשם `tomcat-sw1` מתחת לספריית השורש שלכם שמכילה את כל המידע שהשרת צריך, כולל `servlet` לדוגמה.

אחרי שיצרתם את הספרייה, היכנסו אליה (`cd tomcat-sw1`) והפעילו את השרת. הפעלה וכיבוי של תוכנת השרת מתבצעת בעזרת הפקודות:

init.d-tomcat start

init.d-tomcat stop

על מנת לוודא שבצעתם הכול נכון נסו לגשת לשרת מדפדפן. לצורך הדוגמה נניח ששם השרת שהתחברתם אליו הוא ibmop-05 ושהספרות שבחרתם (XXX שהופיעו קודם לכן) הן 566. הפנו את הדפדפן לכתובת `http://ibmop-05.cs.tau.ac.il:40566/` אם הפעלתם את השרת כשורה, תקבלו דף אינטרנט (ראו תמונה), ואם תקליקו על "a servlet demo" תפעילו את ה-servlet לדוגמה ששמור בקובץ `~/tomcat-sw1/webapps/examples/WEB-INF/classes/HelloWorldExample.java` (יותר נכון בקובץ ה-class של ה-servlet הזה).



הכנת פרוייקט אקליפס

1. הורידו את הקובץ `servlet.jar` מאתר הקורס ושמרו אותו תחת הספרייה `tomcat-sw1` שיצרתם קודם לכן.
2. הורידו את קובץ ה-HTML `address_book.html` מאתר הקורס ושימרו אותו תחת הספרייה `tomcat-sw1/webapps/ROOT`. זהו אינו קובץ מסובך ומומלץ לבחון אותו אם כי אין חובה לעשות כן לצורך מימוש התרגיל.
3. צרו פרוייקט חדש באקליפס, במסך של אשף יצירת הפרוייקט, תנו שם לפרוייקט, ובחרו את האפשרות **Create project from existing source**. בשדה שיפתח כאשר תבחרו את האפשרות הזו, יש לבחור את שם הספרייה שיצרתם עבור השרת - הספרייה `tomcat-sw1` בספרית הבית שלכם (כדאי ללחוץ על הכפתור `Browse` ולבחור את הספרייה הזו מהסייר). כאשר הפרוייקט ייווצר, הוא יכיל כבר את כל הספריות והקבצים שדרושים למימוש ה-`servlet`.

4. Servlet הדוגמה נמצא בספרייה webapps/examples/WEB-INF/classes זוהי גם הספרייה שבה אנו נכתוב את ה servlet שלנו ומחלקות אחרות שנדרשות לצורך עבודתו. מומלץ לבחון את הקוד של הדוגמה.
5. גילשו לדרך http://ibmop.cs.tau.ac.il:40566/address_book.html (בהתאמות של שם המחשב ושל המספר, כמובן). כאשר תקליקו על הכפתורים שאמורים להפעיל servlets חדשים, השרת יחזיר כמובן הודעת שגיאה: ה-servlet שאתם מנסים להפעיל עדיין לא קיים, ולכן tomcat לא יכול להריץ אותו. לשרת שלכם ניתן לגשת לא רק מהמחשב שעליו אתם עובדים, אלא מכל מחשב באוניברסיטה (מחשבים באינטרנט **אינם יכולים לגשת לשרת הזה** מכיון שיש firewall בכניסה לשרת האוניברסיטאית שחוסם את הגישה לתחנות העבודה).
6. יש לתת הוראות מתאימות לקומפיילר. הפעילו את תפריט ההקשר של הפרויקט שיצרתם (עכבר ימני) ובחרו properties. בדיאלוג שיפתח צריך לבחור ראשית את Java Build Path (ברשימה משמאל), ואז את הלשונית Libraries. במידה וה-servlet.jar אינו מופיע ברשימת הספריות הוסיפו אותו. לפני שסוגרים את דיאלוג ה-properties, יש לבחור את האפשרות Java Compiler ברשימה משמאל, לבחור Enable project specific settings, ובשדה של **Compiler compliance level יש לבחור 1.4**. את זה אנו עושים מכיוון שלמרות שהגרסה הנוכחית של ג'אווה היא 6.0, הגרסה של Tomcat שאנו מפעילים משתמשת בגרסה הקודמת של ג'אווה, גרסה 1.4. שימו לב שהשימוש בגרסה מוקדמת זו ימנע מאיתנו שימוש בחלק מהתכונות המתקדמות של Java אותן ראינו בקורס. למשל, תדרשו להתאים את מימוש ספר הטלפונים שכתבתם בתרגיל הקודם כך שלא ישתמש במחלקות גנריות. כל מבני הנתונים שהכרנו עדיין קיימים אך אי אפשר להשתמש בהם בצורתם הגנרית. לדוגמה, במקום `Set<String>` תשתמשו ב `Set` ותדרשו לבצע המרה (casting) לטיפוס המתאים כאשר מוציאים אלמנט מה `Set` כמו כן לא ניתן להשתמש בלולאות `foreach`.

עכשיו אפשר להתחיל לתכנת.



ספר כתובות – מנשק רשת

בתרגיל זה נכתוב ממשק אינטרנטי לספר כתובות. בתרגיל 11 כתבתם מחלקה בשם AddressBook המכילה רשימה של אנשי קשר, ומסוגלת לשמור/לטעון קובץ המכיל ספר כתובות. כעת עליכם ליצור Servlet המאפשר דרך הדפדפן לבצע את הפעולות הבאות:

1. להציג את אנשי הקשר בספר הכתובות (שם בלבד)
2. להציג איש קשר בודד (עם כל הפרטים)
3. להוסיף איש קשר לספר הכתובות (על כל פרטיו)

שלושת השירותים יסופקו ע"י Servlet יחיד, כאשר הצגת הרשימה ואיש הקשר היחיד יתבצעו בעזרת פעולת GET ואילו הוספת איש קשר תתבצע באמצעות POST.

1. הצגת רשימת אנשי קשר

כאשר אתם נתקלים בבקשת Get בלי פרמטרים, עליכם להציג דף אינטרנט המכיל רשימה של אנשי קשר, לדוגמה:

Contact List

- Danny Cohen
- Moshe Vaknin
- Dikla Amzaleg
- Eli Mizrahi
- Eli Ovadia
- Dana Levi

[back](#)

על הדף להראות את רשימת כל אנשי הקשר בספר הכתובות. כל שם ברשימה הינו קישורית (לינק). כאשר נלחץ על הקישורית, הדפדפן יעבור לדף ובו פרטים על איש קשר זה (ראה סעיף הבא).

בנוסף בתחתית הרשימה יהיה לינק back, שבלחיצה על לינק זה נחזור לעמוד הראשי.

2. הצגת פרטי איש קשר

כאשר נתקלים בבקשת Get עם פרמטר יחיד name, עליכם להציג דף ובו פרטי איש הקשר הנ"ל. אם לא קיים איש קשר כזה יש להודיע על כך למשתמש. הדף יכול להראות בסגנון הבא:

Danny Cohen

Email: danny@tau.ac.il
Telephone: 5304466
Address: Taasia 1, Raanana, Israel

[back to homepage](#)

בתחתית הדף יופיע לינק לחזור לעמוד הראשי.

3. הוספת איש קשר חדש

בקשה להוספת איש קשר חדש תתקבל כבקשת POST. בקבלת בקשה כזו ה servlet מוסיף את איש הקשר החדש ומחזיר דף ובו תוצאת ההוספה – הצליח או נכשל (אם איש קשר זה כבר קיים בספר הכתובות) בשני המקרים בתחתית הדף יופיע לינק חזרה לדף הראשי.

הדרכה

התחילו בהסבת קוד ה AddressBook שמימשתם בתרגיל קודם לגרסת ג'אווה 1.4. העתיקו את קובץ הדוגמה HelloWorldExample ושנו את שמו (ואת שם המחלקה) ל AddressBookServlet. הפנו את הדפדפן לכתובת http://ibmop-01:40212/address_book.html והקליקו על View Contacts. אם הכול פועל כשורה תראו שוב את דוגמת ה Hello World. מכאן המשיכו בצעדים קטנים לשנות את קוד הדוגמה ולממש את הפונקציונאליות שהנכם נדרשים לה. למשל, ממשו את doGet ו- doPost כך שידפיסו את הפרמטרים שהתקבלו בבקשה. הוסיפו את החיבור ל AddressBook בצעדים קטנים.

הוראות טכניות

- על ה-Servlet שלכם להיקרא AddressBookServlet
- כאשר אתם מבצעים שינויים בקוד שלכם יש לעצור את שרת ה-tomcat ולהתחילו מחדש.
- ניתן להוריד באתר הקורס כמה דפי HTML סטטיים המופיעים בדוגמאות בסעיפים 1-3. דפים אלו יכולים לשמש אתכם לצורך יצירת דפי ה-HTML ב-servlet. שימו לב שה servlet מייצר דפי HTML באופן דינמי ע"י כתיבת הטקסט ל PrintWriter של אובייקט ה-HttpServletResponse.

הצעות רשות

אם תחשבו על רעיונות נוספים לשלב בתרגיל זה, הרגישו חופשיים לעשות זאת (כל עוד אתם תומכים במינימום הנדרש), לדוגמה

- דפים מעוצבים קצת יותר
- תמיכה בעריכת ומחיקת אנשי קשר
- חיפוש איש קשר לפי טלפון/שם/אימייל
- וכו'

בהצלחה!!