

קורס הנדסת תוכנה – מדריך בניית אב טיפוס

1.....	קורס הנדסת תוכנה – מדריך בניית אב טיפוס	
1.....	הקדמה ומטרות	
1.....	הגשות	
3.....	1. רישום לשירותי רשת	
3.....	1.1. GitHub	
4.....	1.2. Heroku	
4.....	1.3. פורום הקורס	
5.....	1.4. רישום לקורס	
6.....	2. התקנות	
6..	2.1. סביבת פיתוח לרשת בג'אווה – Play Framework 2	
9.....	2.2. לקוח git (Github for Windows)	
9.....	2.3. Heroku Toolbelt	
10.....	2.4. אופציה מומלצת: סביבת פיתוח משולבת IntelliJ	
13.....	3. יצירת אפליקציה בסיסית – Hello JCE	
13.....	3.1. יצירת האפליקציה	
17.....	3.2. עריכת הקוד	
17.....	3.3. בקרת תצורה עם github	
19.....	3.4. הפצה לענן	
19.....	הרחבה: יצירת והעלאת מפתחות הצפנה	
19.....	לסיום: הפצה	

הקדמה ומטרות

במסגרת הקורס נבנה מוצר תוכנה מבוסס טכנולוגיות ווב. בשבועות הראשונים של הקורס, הפרויקט יעבור שלבי אתחול שונים רובם סביב תכנון ומסמכים ובמקביל נבנה בשלבים אב טיפוס עבור המוצר. מטרותיו העיקריות של אב הטיפוס הן:

- היכרות עם טכנולוגיות שיאפשרו ביצוע הפרויקט
 - קבלת משוב מוקדם מהלקוח באמצעות הצגת ממשק המשתמש של המערכת
 - תשתית ראשונית עבור המוצר
 - באופן כללי הנמכת סיכונים לפרויקט
- מדריך זה מפרט את הצעדים המומלצים ליצירת אב הטיפוס בהם רישום לשירותי רשת, התקנות ופירוט ההגשות הנדרשות בכל שלב של אב הטיפוס.

הגשות

ההגשה דרך טופס ההגשות מאתר הקורס. כל שלב יש להגיש עד חצות לפני התרגיל הבא. ההגשה אישית.

להלן השלבים המתוכננים (לא סופי):

שבוע 0 – רישום לשירותי הקורס (עד לתרגיל הראשון)

שבוע 1 – הפצת דף רשת Hello World

שבוע 2 – דף מוצר ראשוני + עיצוב (bootstrap)

שבוע 3 – אב-טיפוס ממשק משתמש (במסגרת משימת פרויקט SRS)

שבוע 4 – צד שרת בסיסי

שבוע 5 – השלמת אב טיפוס (במסגרת הגשת פרויקט ZFR)

1. רישום לשרותי רשת

במסגרת הקורס נעשה שימוש במספר שרותים ברשת, בפרט github, heroku ופורום לקורס.

1.1 GitHub

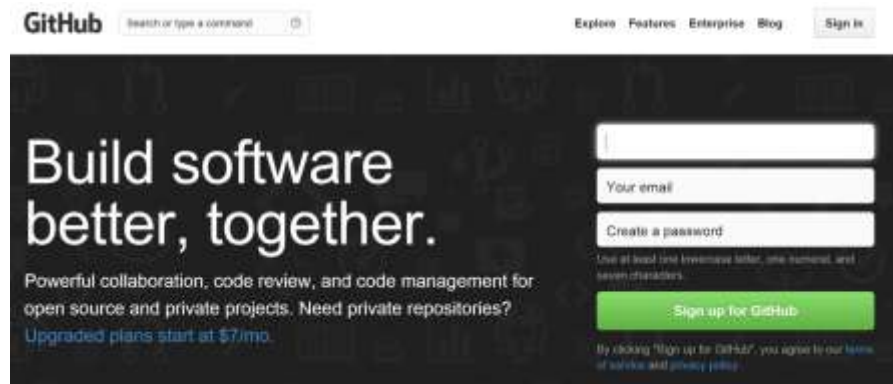
רקע

Git היא מערכת בקרת גרסאות (נרחיב בנושא בהמשך הקורס) שפותחה במקור עבור פיתוח לינוקס. [GitHub](#) הוא שרות מבוסס git בענן וכן מערכת ניהול פרויקטי תוכנה. בקורס נשתמש ב- github בצורה נרחבת.

פתיחת חשבון

עליכם לפתוח חשבון ב github:

לכו ל**עמוד הבית**:



והירשמו לשרות ע"י בחירת שם משתמש, מייל וסיסמא.

הערה חשובה: ככל הנראה עכשיו, שם המשתמש שתבחרו ילווה אתכם עוד כברת דרך כמהנדסי תוכנה ואפילו משמש **כמעין CV** **בחלק מהמקומות**, כך שכדאי לשקול אם אתם מעוניינים להשתמש בכינוי או בשמכם האמיתי.

בעמוד הבא אין צורך לבחור בתכנית בתשלום... (שמאפשרות למשל פתיחת אתרי פרויקט פרטיים) ורק לאחר את יצירת החשבון. לאחר הרישום אפשר להוסיף פרטים שונים ב**עמוד החשבון** לבחירתכם.

אתם נדרשים לפחות למלא בשדה האחרון מייל שמקושר לתמונה לאוואטאר שלכם (באמצעות השרות [Gravatar](#) - ברישום נפרד). באופן זה יהיה קל יותר לנהל בהמשך את משימות הפרויקט השונות.

jce-test-student

- Profile
- Account Settings
- Emails
- Notification Center
- Billing
- Payment History
- SSH Keys
- Security History
- Applications
- Repositories
- Organizations

PRIVATE REPOS 0 OF 0

Public Profile


Name

Email (will be public)


URL

Company

Location

Gravatar Email (Private) 

jce-test-student@post.jce.ac.il

 [Change your avatar at Gravatar.com](#)

את שם המשתמש יש לשלוח לצוות הקורס בהקדם, כדי שתקבלו הרשאת כתיבה באתר הקורס הנצרכת כבר למשימת הפרויקט הראשונה שלכם.

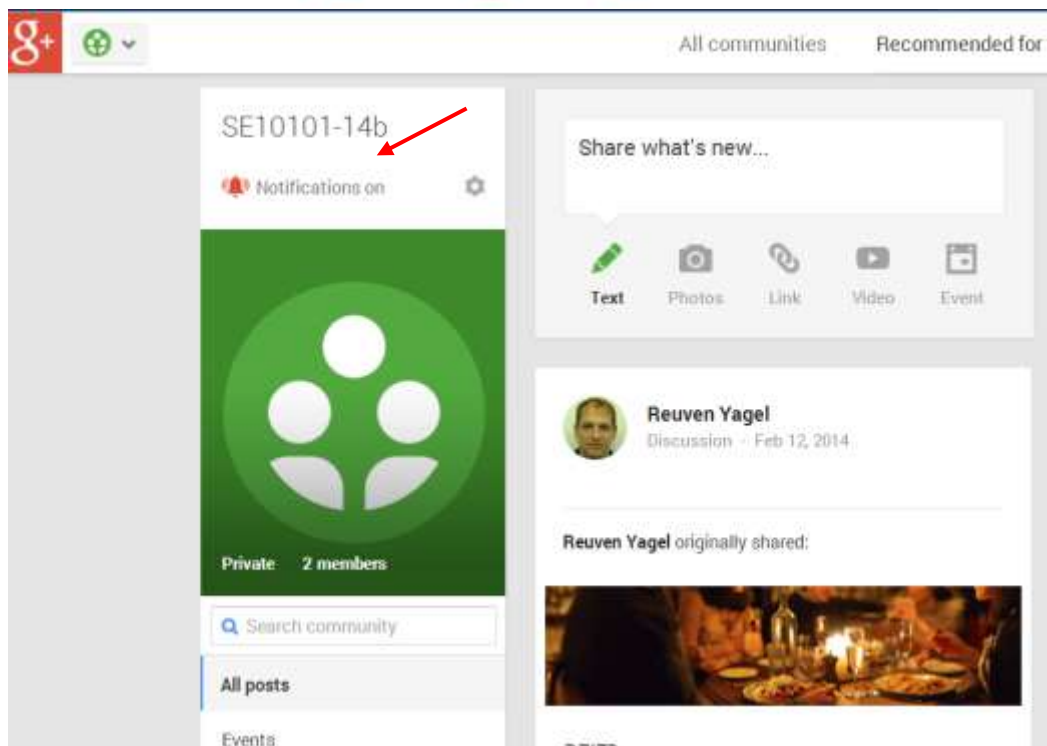
2.1 Heroku

Heroku הוא שרות הפצה (delivery) לאפליקציות בענן (למתעניינים: בשיטה הנקראת [Platform as a Service](#) – PaaS). השרות ישמש אתכם להפצת והרצת אב-טיפוס שלכם בשלביו השונים ובהמשך עבור המוצר שתפתחו בפרויקט. באופן כזה החל מהיום הראשון לפיתוח יש לכם לאן להפנות לקוחות מתעניינים ואתם יודעים שהמוצר שלכם במצב פעיל.

כרגע כל שנדרש הוא לפתוח חשבון חינמי ע"י [רישום](#) באמצעות המייל (מומלץ אותו אחד שמסרתם ל-github) והפעלה במייל שתקבלו מהם. הערה: אפשר להמתין עם הרישום עד להגשת אב-טיפוס 1.

3.1 פורום הקורס

כדי להישאר מעודכנים ולשתף פעולה בצורה מיטבית, פתחנו קהילה של google+. מומלץ להגדיר עדכון במייל או התראות בטלפון/שולחן עבודה על הודעות כך שתהיו מעודכנים בנעשה. ניתן לעשות זאת דרך הלינק [והגדרת תזכורות](#) כפי שמתואר בתמונה הבאה:



אם תוסיפו תמונה שלכם זה גם יעזור לתקשורת בקורס.

4.1. רישום לקורס

שלחו לנו (בפורום) את פרטיכם
נא לכלול:

- שם
- ת.ז.
- מייל
- שם משתמש ב-github

עד כאן לתחילת הקורס (משימה אישית 0) - נא להקדים ברישום כדי לקבל הרשאות כתיבה באתר הקורס.

2. התקנות

בכדי לפתח את אב הטיפוס נידרש למספר ספריות ותוכנות תוכנות אלו כבר מתוקנות במעבדות המכללה (בעתיד ייתכן ונספק מכונה וירטואלית מוכנה – כדי לחסוך את הצורך בהתקנות).

1.2. סביבת פיתוח לרשת בג'אווה – Play Framework 2

Play היא סביבת פיתוח מודרנית לאפליקציות רשת בג'אווה.

להלן הצעדים להתקנת Play בחלונות 7 / 8.1, (בלינק ישנן גם הוראות התקנה למערכות אחרות). מובאים כאן מקורות נוספים ללימוד סביבה זו.

1.1.2. תנאי קדם: התקנת JDK מגרסה 6 ומעלה.

הערה: מומלץ להתחיל משלב 2, וידוא ההתקנה, ולבצע את שלב 1 רק אם ה-JDK לא מותקן על המחשב.
1. יש לבחור מהקישור את הגרסה המתאימה למערכת שעליה מתקינים.

ייתכן שתצטרכו גם להוסיף ל-PATH את ספריית ההתקנה (bin) של ה-JDK ולהיכנס מחדש למחשב (log on):

- הוראות להוספת Path (במקלדת Win+Break):
-

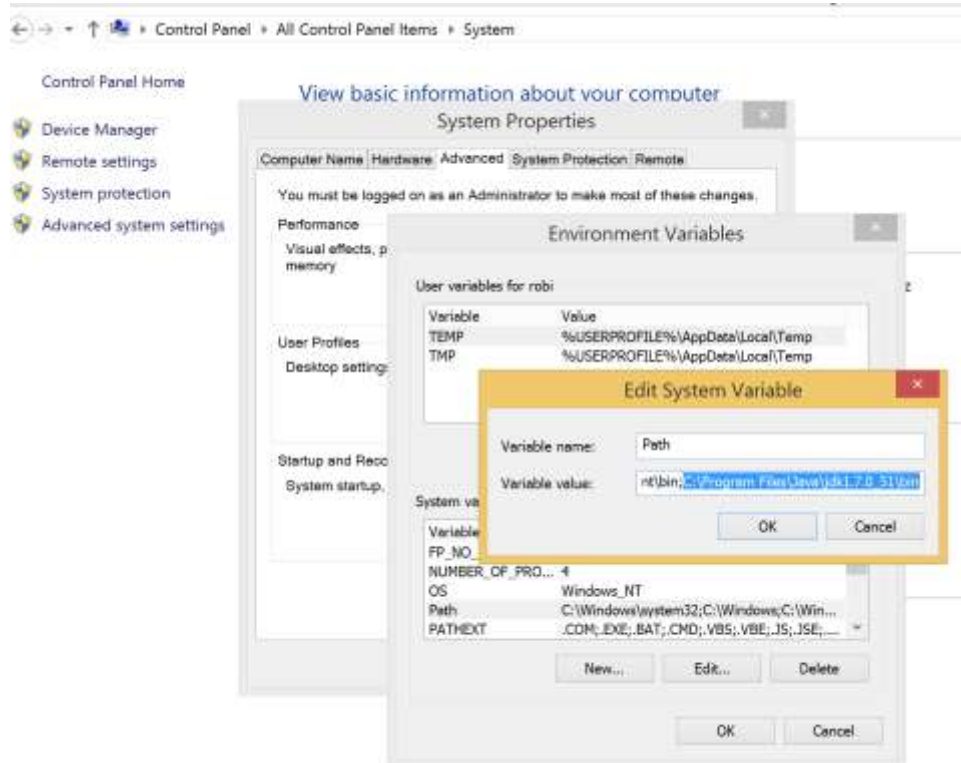
Control Panel > System > Advanced System Settings >

Advanced Tab > Environment Variables >

System Variables box > Path > click on Edit > Variable Value ->

הוסיפו בסוף הטקסט ';' ואחריו את הנתיב ל ספריית הbin של הjdk (בד"כ משהו כמו

(C:\Program Files\Java\jdk1.7.0_10\bin\



2. וידוא התקנה: פתח מעטפת (CMD :Shell או PowerShell) והרץ

- Java -version
- javac -version

```

Select Windows PowerShell

C:\Users\robi> java -version
java version "1.7.0_45"
Java(TM) SE Runtime Environment (build 1.7.0_45-b18)
Java HotSpot(TM) Client VM (build 24.45-b08, mixed mode, sharing)
C:\Users\robi> javac -version
javac 1.7.0_51
C:\Users\robi>
  
```

הערה: אם אין לכם הרשאות לשינוי משתנה סביבה במערכת (למשל במחשב בעבודה) עדיין תוכלו לשנות רק עבור המשתמש הנוכחי.

2.1.2. התקנת play

גשו לעמוד ההורדות <http://www.playframework.com/download>

והורידו בצד ימין את הגרסה האחרונה של play (כרגע 2.2.2 - ישנה שם אפשרות להוריד את typesafe שהיא חבילה מורחבת יותר – אנחנו נסתפק באופציה של Classic Distribution).

את קובץ ה-ZIP פתחו והתקינו במקום כרצונכם, למשל c:\play-2.2.2 והוסיפו מסלול זה ל-PATH כפי שהוסבר לגבי Java בתנאי הקדם.

אם יש צורך ראו הוראות התקנה מפורטות כאן:

<http://www.playframework.com/documentation/2.2.x/Installing>

וידוא התקנה: פתחו מעטפת והריצו play, צריך להתקבל מסך כדלקמן:

```

Windows PowerShell

C:\Users\robi> play
Getting org.fusesource.jansi jansi 1.11 ...
:: retrieving :: org.scala-sbt#boot-jansi
  confs: [default]
  1 artifacts copied, 0 already retrieved (111kB/15ms)
Getting com.typesafe.play console_2.10 2.2.2 ...
:: retrieving :: org.scala-sbt#boot-app
  confs: [default]
  6 artifacts copied, 0 already retrieved (2012kB/106ms)
Getting Scala 2.10.3 (for console)...
:: retrieving :: org.scala-sbt#boot-scala
  confs: [default]
  5 artifacts copied, 0 already retrieved (24447kB/1013ms)

play

play 2.2.2 built with Scala 2.10.3 (running Java 1.7.0_45), http://www.playframework.com
This is not a play application!
Use 'play new' to create a new Play application in the current directory,
or go to an existing application and launch the development console using 'play'.
You can also browse the complete documentation at http://www.playframework.com.
C:\Users\robi>
  
```


2.2. לקוח git (Github for Windows)

בכדי לנהל גרסאות עם git ו-github, יש להתקין תוכנת לקוח במחשבכם. התקנה זו נצרכת כבר להגשת אישית ראשונה.

במידה ואתם עובדים עם מערכת הפעלה windows, הדרך הפשוטה ביותר להתקנה היא התקנת הלקוח של github מכאן <http://windows.github.com> התקנה זו מאפשר ניהול המאגרים שלכם בתוכנת חלונות מול github וגם מקומית במחשב (בנוסף מותקנת גרסת מעטפת (shell) מותאמת לעבודה עם git וכן טיפול במפתחות הצפנה).

וידוא התקנה: ייפתח חלון התוכנה GitHub

הערה: אפשר לחלופין להתקין רק את git מכאן <http://git-scm.com/downloads> בנוסף ישנו את [egit](#) שהוא לקוח git ל-eclipse. באמצעותו תוכלו לנהל גרסאות ישירות מסביבת העבודה. גם בסביבת העבודה IntelliJ יש לקוח מובנה.

הערה: windows יש לוודא שהותקן גם (בשולחן העבודה) [git-bash](#) (אם לא, ניתן להוריד מהלינק).

3.2. Heroku Toolbelt

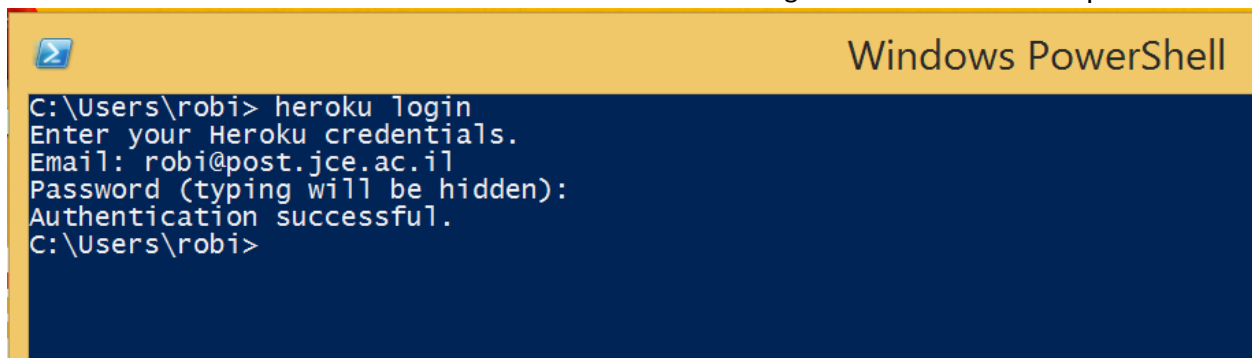
Heroku הוא שרות הפצה (delivery) לאפליקציות בענן בשיטה הנקראת [Platform as a service](#) – PaaS. השרות יישמש אתכם להפצת והרצת אב-הטיפוס שלכם בשלביו השונים ובהמשך עבור המוצר שתפתחו בפרויקט. באופן כזה החל מהיום הראשון לפיתוח יש לכם לאן להפנות לקוחות מתעניינים ואתם יודעים שהמוצר שלכם במצב פעיל.

אנו נשתמש בכלי Heroku Toolbelt על מנת לבצע את ההפצה והניהול של האפליקציה לענן (Heroku). זהו סט כלים מ-Heroku שבאמצעותם מתבצעת ההפצה והניהול של מוצר התוכנה לענן מתוך המחשב שלכם.

ניתן להוריד את קובץ ההתקנה מכאן <https://toolbelt.heroku.com/windows> (הרצת הקובץ תתקין גם תוכנות נוספות, ביניהן git).

להוראות התקנה מפורטות ראו גם כאן: <https://devcenter.heroku.com/articles/quickstart>

ודאו את ההתקנה ע"י הרצה במעטפת: heroku login



```
Windows PowerShell

C:\Users\robi> heroku login
Enter your Heroku credentials.
Email: robi@post.jce.ac.il
Password (typing will be hidden):
Authentication successful.
C:\Users\robi>
```

ייתכן שיוצע לכם להעלות מפתחות לשירות. אם לא יש להריץ את הפקודה:
heroku keys:add

פקודה זו תיצור בספריית המשתמש שלכם זוג מפתחות הצפנה אם לא קיימים ותעלה אותם ל-heroku.

4.2. אופציה מומלצת: סביבת פיתוח משולבת IntelliJ

אפשר לעבוד עם play עם כל עורך פשוט, אך יש תמיכה טובה ב-IntelliJ.

הערה: אנו נראה את השימוש ב-IntelliJ בקורס זה מכיוון שהוא הכי פשוט לקינפוג אתם יכולים להשתמש בכל סביבה אחרת, אך את הוראות ההפעלה תצטרכו למצוא בכוחות עצמיכם.

1. התקנה:

- a. התקינו את Play לפי ההוראות [האלו](#)
- b. הורידו את קובץ ההתקנה מפה: <http://www.jetbrains.com/idea/download>
- c. התקינו את הקובץ. כשתבקשו להזין רשיון, הכניסו את הרשיון של המכללה:

- i. שם משתמש: Jerusalem College of Engineering
- ii. License key: יפורסם בפורום הקורס

הערה: לחצו Next עד שתגיעו למסך פתיחת פרויקט.

- d. בשלב זה ניצור את הקבצים שיאפשרו לפתוח את הפרויקט הפרויקט ב-IntelliJ:
 - i. הכנס לתיקיית הפרויקט ב-Command line
 - ii. הכנס ל Console של Play ע"י הקלדת Play
 - iii. הקלד את הפקודה Idea .

```

C:\>cd test
C:\test>play
[info] Loading project definition from C:\test\project
[info] Set current project to test (in build file:/C:/test/)

play 2.2.1 built with Scala 2.10.2 (running Java 1.7.0_51), http://www.playframework.com

> Type "help play" or "license" for more information.
> Type "exit" or use Ctrl+D to leave this console.

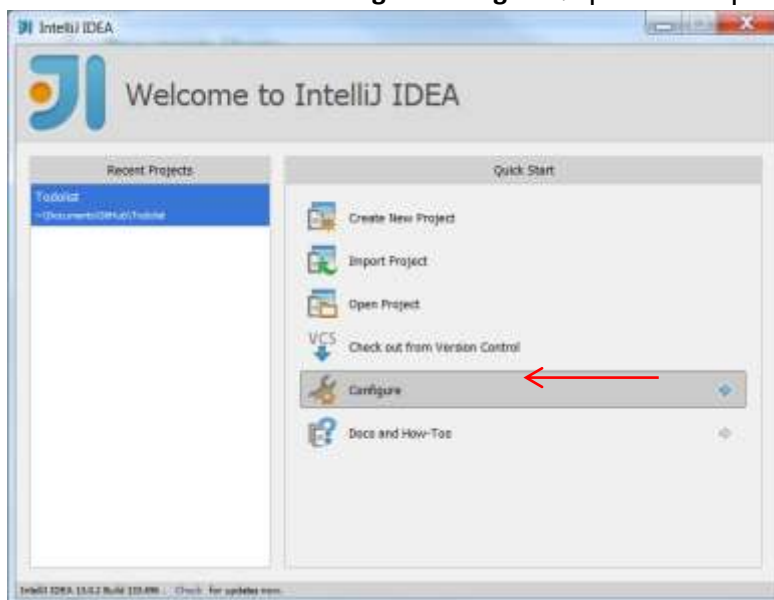
[test] $ idea
[info] Creating IDEA module for project 'test' ...
[info] Running compile:managedSources ...
[info] Excluding folder target
[info] Created C:\test\.idea\IdeaProject.iml
[info] Created C:\test\.idea
[info] Excluding folder C:\test\target\scala-2.10\cache
[info] Excluding folder C:\test\target\scala-2.10\classes
[info] Excluding folder C:\test\target\scala-2.10\classes_managed
[info] Excluding folder C:\test\target\native_libraries
[info] Excluding folder C:\test\target\resolution-cache
[info] Excluding folder C:\test\target\scala-2.10\resource_managed
[info] Excluding folder C:\test\target\streams
[info] Created C:\test\.idea_modules\test.iml
[info] Created C:\test\.idea_modules\test-build.iml
[test] $
    
```

2. פתיחה IntelliJ:

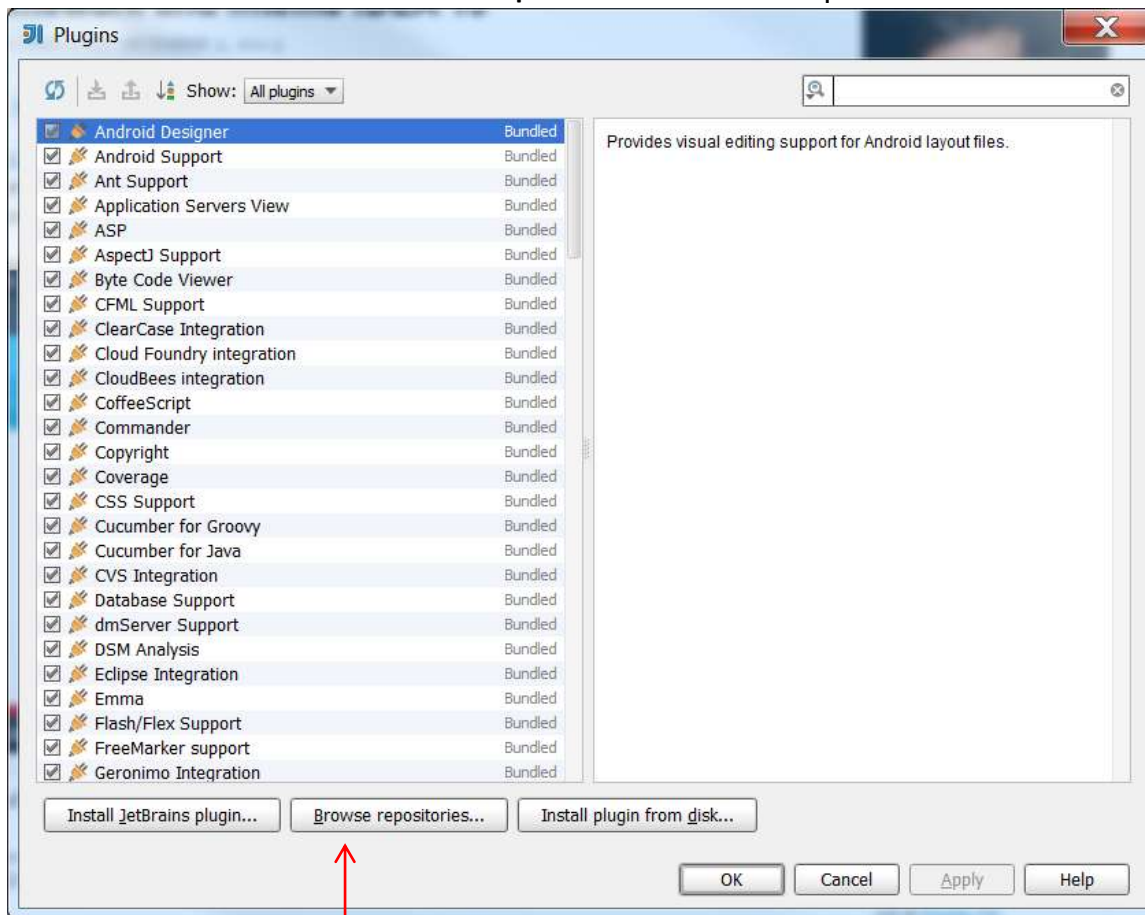
- א. פתח את IntelliJ.
- ב. עקבו אחר ההוראות...

a. כאשר תתבקשו לבחור מערכת בקרת גרסאות (version control system) ודאו ש-git מסומן

ג. בחלון שנפתח לחץ על **Configure > Plugins**



ד. בחלון שנפתח לחצו על **Browse repositories**



- ה. בחלון שנפתח חפשו את **play 2.0 support** בתיבת הפילטר ואשרו.
- ו. חזרו על שלבים ב-ד אך הפעם חפשו את הרשומה של **Scala**.

3. יצירת אפליקציה בסיסית – Hello JCE

1.3. יצירת האפליקציה

בדף [זו](#) תוכלו למצוא הוראות מפורטות בנוגע לבניית אפליקציה בסיסית כפי שמייד נעשה, להלן מקורות להרחבה שאולי תזדקקו להם בהמשך הסמסטר:

מקורות

- ספר אלקטרוני Play Framework Cookbook, זמין באתר הספרייה (לקריאה ברשת המכללה, בבית צריך קודם להיכנס דרך אתר הספרייה ל"גישה מהבית")
<http://proquest.safaribooksonline.com/9781849515528>
 - (מאגר זה מכיל גם ספר מקוצר יותר)
- ספר נוסף רלוונטי Play for Java
- עמוד התיעוד הראשי <http://www.playframework.com/documentation/2.2.x/JavaHome>
- אפליקצית TODO לדוגמא
<http://www.playframework.com/documentation/2.2.x/JavaToDoList>
- גרסה עם צד לקוח מודרני יותר
<http://www.playframework.com/documentation/2.2.x/JavaGuide1>

הוראות

1. יצרו את התכנה מתוך המעטפת (למשל בדוגמא זו HelloJce הוא שם התכנית) :

```
$ play new HelloJce
```

2. לחצו Enter או בחרו שם לאפליקציה (Enter יבחר את השם שהזנתם קודם).

3. בחרו 2 לאפליקציית Java.

```
C:\>play new test

play 2.2.1 built with Scala 2.10.2 (running Java 1.7.0_45), http://www.playframework.com
The new application will be created in C:\test
What is the application name? [test]
>
Which template do you want to use for this new application?
  1 - Create a simple Scala application
  2 - Create a simple Java application
> 2
OK, application test is created.
```

4. כנסו לתיקיה שנוצרה cd HelloJce

5. הריצו את האפליקציה ע"י play ~run

```

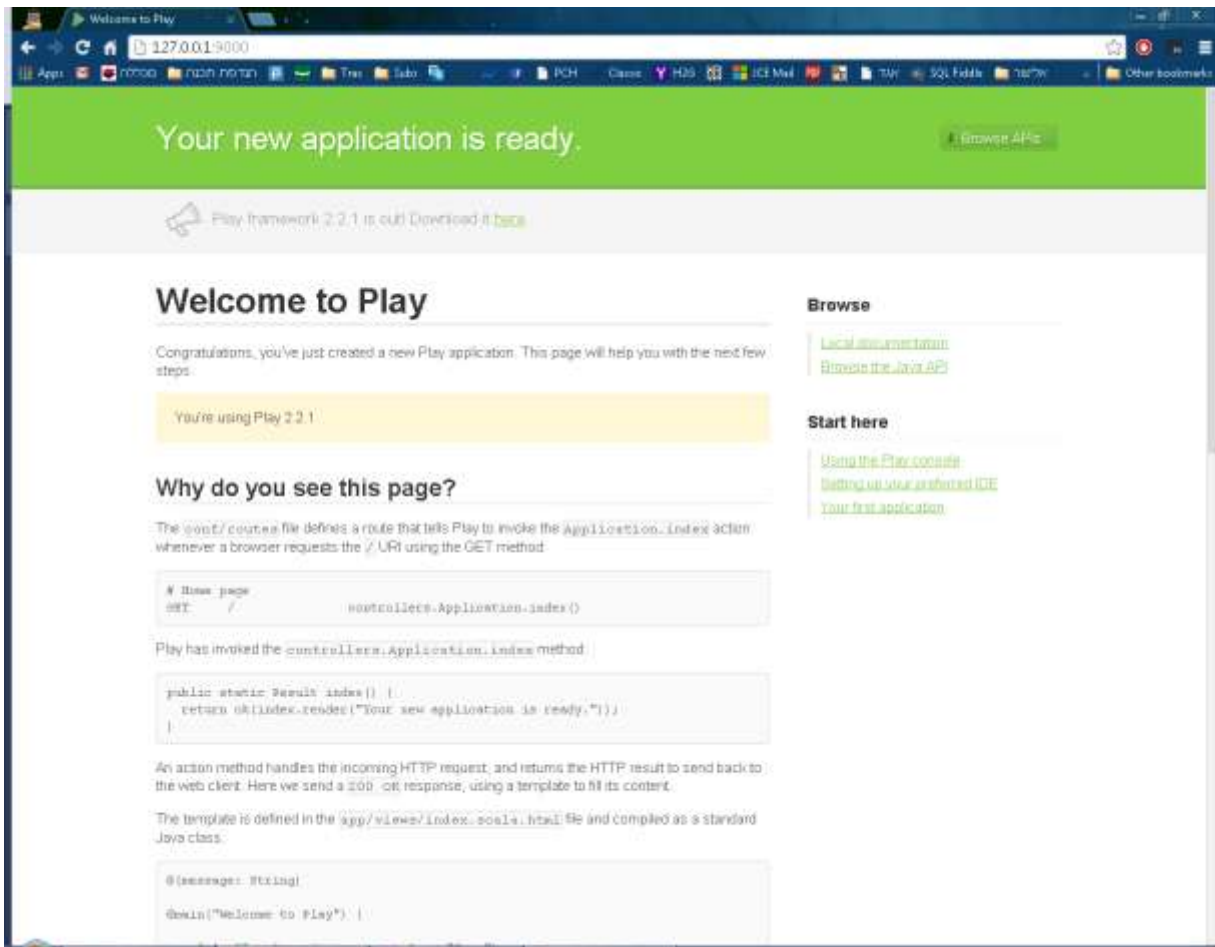
Administrator: C:\Windows\system32\cmd.exe - play run

C:\>\cd test
C:\test>play run
[info] Loading project definition from C:\test\project
[info] Set current project to test (in build file:/C:/test/)

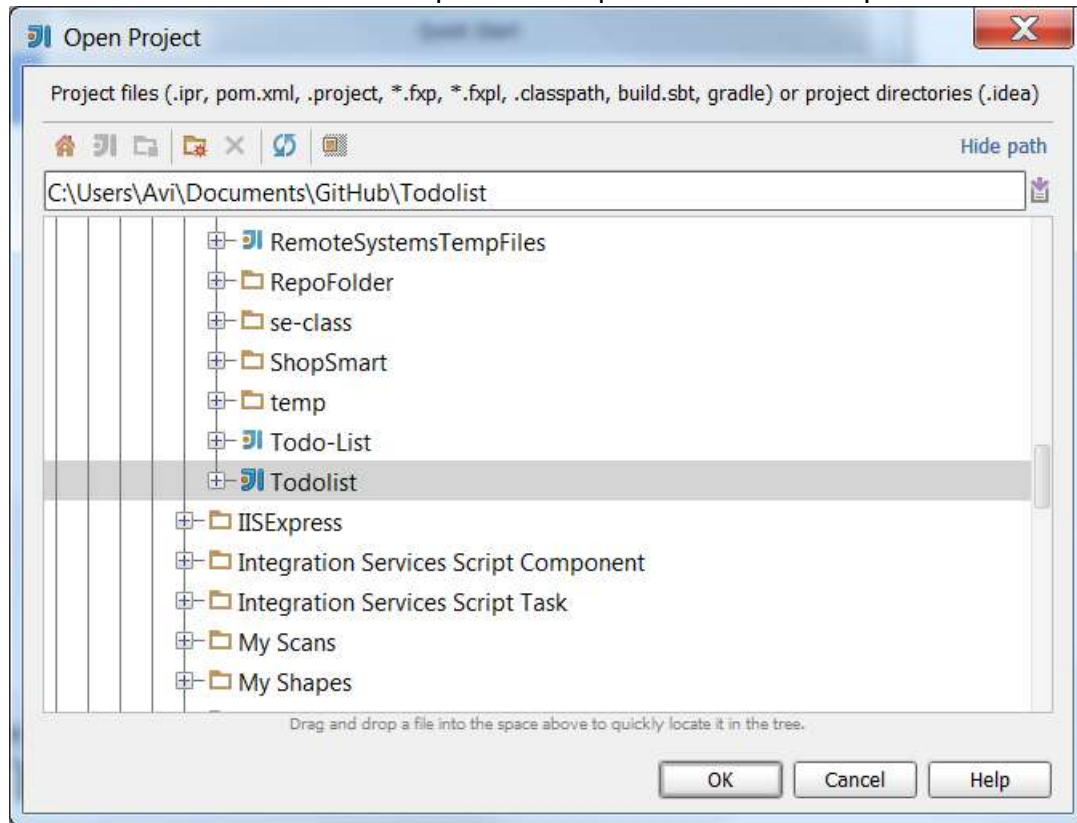
--- <Running the application from SBT, auto-reloading is enabled> ---

[info] play - Listening for HTTP on /0:0:0:0:0:0:0:0:9000
<Server started, use Ctrl+D to stop and go back to the console...>
[info] Compiling 4 Scala sources and 2 Java sources to C:\test\target\scala-2.10\classes...
[info] play - Application started (Dev)
  
```

6. ודאו שאת מסוגלים לראות את האפליקציה באמצעות דפדפן אינטרנט בכתובת
127.0.01:9000



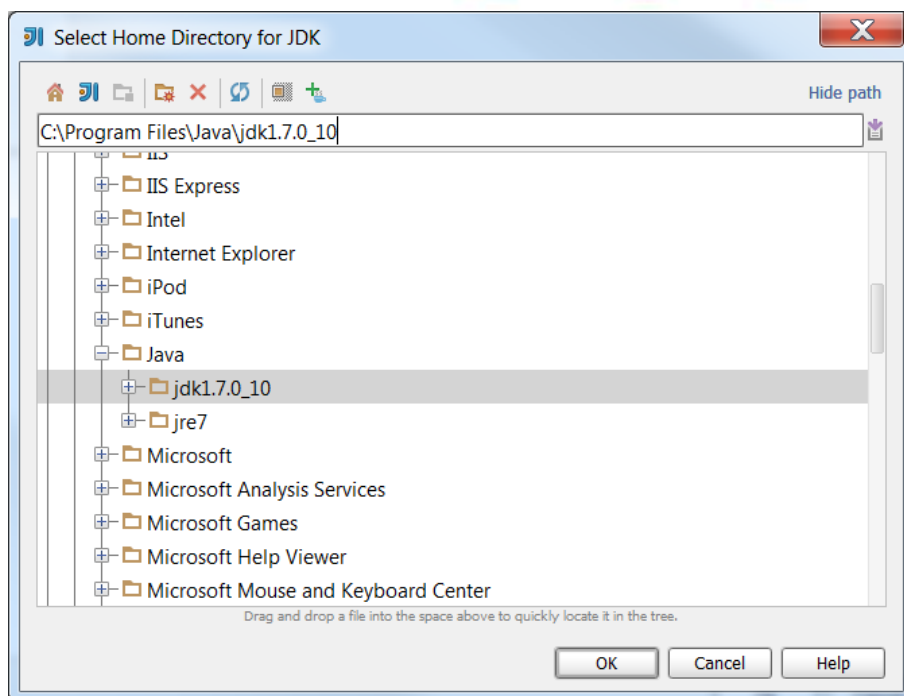
7. אופציה: פתיחת הפרויקט ועבודה ב- IntelliJ (בשלב זה מספיק גם עורך טקסט פשוט, כך שניתן גם לדלג לסעיף הבא של עריכת הקוד 3.2)
- יצירת קבצי פרויקט: במעטפת בתיקיית האפליקציה – הפעילו: play idea
 - במסך הפתיחה של IntelliJ בחרו **open project**.
 - בחלון שנפתח הזינו את התיקייה של הפרויקט שלכם.



- בשלב זה אמורה להיפתח סביבת הפיתוח.
- ייתכן ותראו שיש צורך לקנפג את הntib לאדן

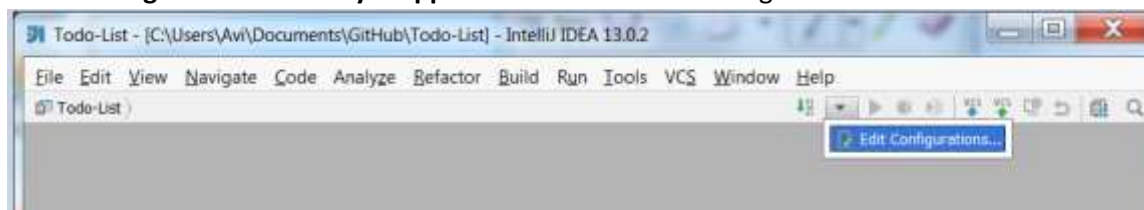


- אם זה המצב, לחצו על **JDK > + > Setup SDK > Configure** והזינו את הntib לאדן שעל המחשב שלכם. (אני נעזרתי בתכנת החיפוש [Everything](#) ובחיפוש אחר המחרוזת JDK נכדי למצוא את הntib במחשב שלי)



ז. הוסיפו קונפיגורציה לקומפיילר:

Edit configuration > + > Play 2 App > Give a name to the configuration > OK



- ח. אם תרצו תוכלו לשנות את קיצורי המקלדת לסביבה מוכרת יותר דרך
File > Settings > keymap > keymaps > (Visual studio / eclipse/ ...)
 ט. הריצו את הפרויקט (יפתח הדפדפן עם עמוד האינטרנט)

2.3. עריכת הקוד

מרשימת קבצי הפרויקט פתח את app <- controllers <- Application.java ושנה את המחרוזת "Hello..." הנשלחת לתצוגה להודעה משלך, שמור ורענן את הדפדפן



8. כדי לאפשר בהמשך הפצה על גבי Heroku לאפליקציה עם DB, הוסיפו לקובץ application.conf בתיקייה conf את השורה הבאה (למען הסדר הטוב מומלץ למקם תחת הקטגוריה Evolutions):

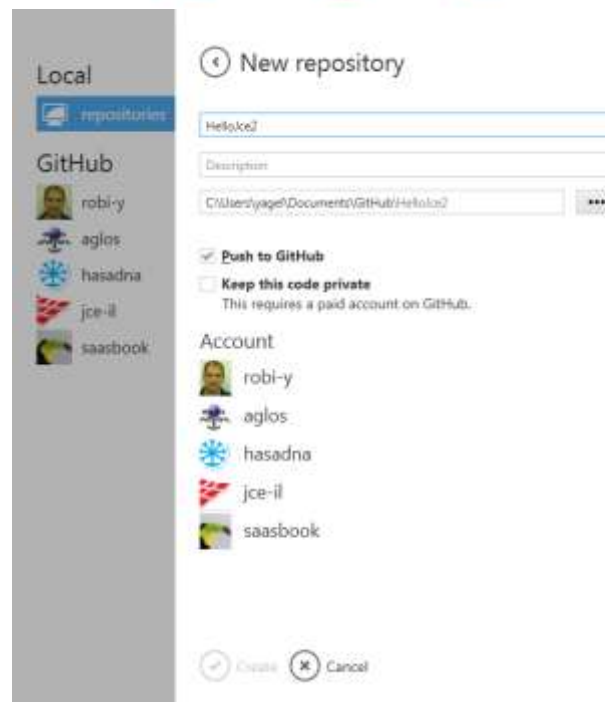
```
applyEvolutions.default = true
```

3.3. בקרת תצורה עם github

אחרי שיצרתם את האפליקציה הבסיסית, עליכם להוסיף קבצים אלו למערכת ניהול הגרסאות github. המטרה היא שהקוד והגרסאות שלו ינוהלו עם בקרת תצורה ובשרת נגיש (נלמד בהמשך) וכך תוכלו בין השאר לשתף פעולה באופן יעיל עם מפתחים אחרים.

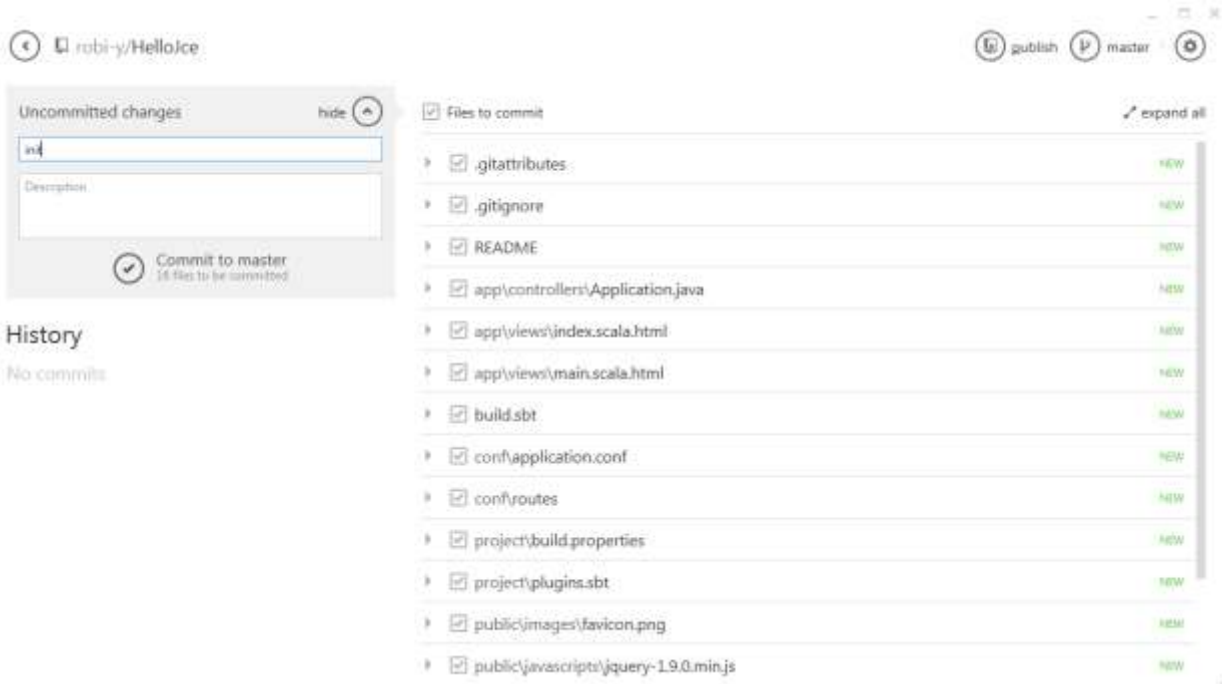
להלן ההוראות באמצעות הלקוח Github for windows (לתרחישים מורכבים יותר נלמד בהמשך בפירוט איך לעבוד עם git ו-github).

1. פתחו את הלקוח ובחרו בתפריט +Create
2. בחרו שם למאגר בשם של האפליקציה שלכם ובמיקום את הספרייה שמעליה (השאירו מסומן את האופציה push to github)



3. לחצו על Create

4. במסך הראשי הקליקו על המאגר לפתיחת פרטיו



5. שמירת גרסה מקומית: תחת Uncommitted Changes הכניסו הערה, למשל init

לחצו על Commit to master

6. שליחה ל-github: לחצו על כפתור publish למעלה

7. וידוא: מתפריט ההגדרות מימין למעלה, בחרו View on Github. המאגר אמור להיפתח בדפדפן עם רשימת הקבצים. הכתובת שנפתחה היא חלק א של ההגשה.

4.3. הפצה לענן

צור אפליקציה ב-heroku:

```
$ heroku create
Creating infinite-dawn-1951... done, stack is cedar
http://warm-1289.herokuapp.com/ | git@heroku.com:warm-1289.git
Git remote heroku added
```

צעד זה מאתחל אפליקציה בשרות (במקרה זה בשם infinite-dawn-1951, אפשר גם לתת שם כפרמטר: `Heroku create {app-name}`) – מומלץ בהמשך בשביל לתפוס שם שתואם למוצר שלכם, וגם מקשר ענף git עבור הפצת הקוד.

הרחבה: יצירת והעלאת מפתחות הצפנה

אם לא ביצעתם בעת התקנת heroku toolbelt למעלה להלן הרחבה בנושא: צעדים אלו נצרכים פעם אחת בכל מחשב. אפשר גם להשתמש במפתחות שכבר יש לכם בהנחה שהעברתם אותם לתיקיית ssh. בפרופיל המשתמש.

שים לב לשמות התיקיות המתאימים במקום מה שב<סוגריים>. תתבקש גם להכניס סיסמא עבור המפתח, בחר משהו שתוכל לזכור בכל שימוש עתידי במפתח.

שימו לב, אתם לא נדרשים להבין את כל המהלך הבא, אך כן לדעת לעקוב אחרי ההוראות ע"מ לאפשר את התקשרות עם Heroku

הערה: כל נושא המפתחות מצריך לימוד, עזרה אפשר למצוא למשל [כאן](#), [כאן](#) ו**כאן**, שימו לב שההפצה ב-ssh ואינה חסומה ע"י firewall ברשת אליה אתם מחוברים – במקרה כזה אפשר קודם לשמור ב-github (סעיף 2.2) ולהפיץ מאוחר יותר. כנראה המפתחות שיצרה התקנת github for windows נדחים בגלל חוסר ב-passphrase.

הערה: ב-windows יש להריץ את הסקריפטים הבאים מ git-bash

```
$ heroku login
$ cd C:\Users\<user>\.ssh
$ ssh-keygen -t rsa -f id_rsa
Enter passphrase...
$ heroku keys:add id_rsa.pub
$ cd <back to app dir>
```

לסיום: הפצה

לסיום, הפצת האפליקציה (בתלות בהגדרות אולי תתבקש להזין את הסיסמא של המפתחות), התהליך לוקח מספר דקות שבהם heroku מכינים מכונה וירטואלית עבור האפליקציה:

```

$git push heroku master
Counting objects: 34, done.
Delta compression using up to 8 threads.
Compressing objects: 100% (28/28), done.
Writing objects: 100% (34/34), 970.15 KiB, done.
Total 34 (delta 0), reused 0 (delta 0)

-----> Play 2.x app detected
...
    http://8044.herokuapp.com deployed to Heroku
To git@heroku.com: infinite-dawn-1951.git
* [new branch]      master -> master
  
```

וידוא שהאפליקציה רצה:

בדוק בדפדפן בכתובת של הפצת האפליקציה, לדוגמא:
[http:// infinite-dawn-1951.herokuapp.com/](http://infinite-dawn-1951.herokuapp.com/)



לקבלת פרטים נוספים על ריצת האפליקציה ניתן להריץ את הפקודות:

```

$ heroku ps
$ heroku log
  
```

עד כאן לשבוע 1.

להגשה – משימה אישית 1a - שני קישורים:

1. לאפליקציה הרצה ב-heroku

2. למאגר הקוד ב- github