**תרגיל בית 8 (עבודה מקדימה)**

תרגיל זה יוקדש ל-Software Frameworks. נציג לכם שני מושגים חשובים עבור הנדסת תוכנה: הזרקת תלויות, DI (לא לבלבל עם DIP) ותכנות מונחה היבטים, AOP.

על מנת להשתמש במושגים אלה, ניקח את התשתית מ-Framework שתומך בהם ושכולל את כל הפונקציונליות הבסיסית הדרושה.

הדרך הקלה ביותר להשתמש ב-Framework הנחוץ היא להתקין Intellij Ultimate. תוכלו לקבל [רישיון חינם](https://www.jetbrains.com/student/) מכיוון שאתם סטודנטים. כל מה שנחוץ הוא כתובת מייל של JCT. במודל יש הסבר מפורט לגבי התקנת התכנה.

## התקנת AspectJ

1. התקינו את aspectj באמצעות לחיצה כפולה על קובץ ה-jar המסופק לכם (אם הקובץ לא נפתח, עליכם להתקין Java על המחשב https://www.java.com/en/download/manual.jsp). **זכרו את מיקום ההתקנה**. נסמן את מיקום ההתקנה ב-<AspectjDir>. ברירת המחדל למיקום זה הינו c:/aspectj1.9
2. [שנו את משתנה הסביבה PATH](https://java.com/en/download/help/path.html)  שלכם כך שיכלול <AspectjDir>**/bin**.

## הגדרת Aspectj ו-DI ב-Intellij

1. התקינו את ה plugin של Aspectj באינטליג'יי:
   1. Settings | Plugins | Marketplace | Type "AspectJ" | Install
   2. Restart Intellij
2. New | project | Jakarta EE
3. בתפריט Project template בחרו Web application
4. לחצו על Next
5. check CDI and Weld SE from Dependencies
6. בתצוגת הפרוייקט (בצד שמאל של המסך לאחר פתיחת הפרוייקט) פתחו את הקובץ src/main/resources/META-INF/beans.xml ושנו בו את ערך השדה bean-discovery-mode מ "annotated" ל "all" (במידה והשדה לא קיים, הוסיפו אותו – ראו צילום מסך למטה).
7. הוסיפו <AspectjDir>/lib/aspectjrt.jar (שנוצר ברגע שאתם מתקינים aspectj) כתלות של הפרוייקט שלכם באופן הבא (ראו צילום מסך למטה):
8. file | project structure | modules | dependencies | + | Jars | aspectjrt.jar
9. שנו את הקומפיילר ל-ajc באופן הבא (ראו צילום מסך למטה):
10. File | Settings| Build, Execution, Deployment | Compiler | Java Compiler.
11. השתמשו ברשימת הקומפיילרים, בחרו ajc.
12. הגדירו את המסלול ל-<AspectjDir>/lib/aspectjtools.jar
13. שנו את Target Bytecode version ל 13

[בקישור הבא](https://drive.google.com/drive/folders/1rgcUJN7fZShR4ZnzVIqITpx26-PFd-hd?usp=sharing) תוכלו למצוא הקלטות של ביצוע הצעדים לעיל (שימו לב שלא כל הצעדים מופיעים בהקלטה. עליכם לבצע את כל הצעדים)

בנוסף, מצורפים צילומי מסך של צעדים עיקריים

A computer screen shot of a computer code

AI-generated content may be incorrect.

A screenshot of a computer

AI-generated content may be incorrect.

A screenshot of a computer

AI-generated content may be incorrect.

# נספח 1: מענה לשאלות ושגיאות נפוצות בתרגיל 8

* אנא ודאו שאתם עובדים עם גרסת Intellij Ultimate ולא הגרסה החינמית. אין בגרסה החינמית תמיכה ב Java EE.
* אנא עקבו אחרי ההוראות בסרטונים באופן מדוייק
* במהלך הריצה עם Weld יש הודעות באדום והודעה בסוף עם Weld shut down. אין כאן שגיאה - זהו הפלט הצפוי כשמריצים את ה Framework
* יש מספר סטודנטים שהשימוש ב Named לא עובד עבורם. ראינו בכיתה אלטרנטיבה הדורשת הגדרת אנוטציה מותאמת אישית. השתמשו בה.
* כאשר אתם מגדירים אנוטציה מותאמת אישית זכרו להגדיר אותה עם האנוטציות הנלוות: Qualifier ו Retention.
* כאשר אתם משתמשים ב Producer הצהירו על כך שהוא מחזיר SortingAlgorithm<Integer> ולא רק SortingAlgorithm
* אל תשנו דבר בקבצים של אלגוריתמי המיון. אינכם מגישים אותם.
* הרבה סטודנטים מנסים כל מיני פתרונות ומאוחר יותר כל הקוד שלהם מלא בשאריות מניסויים קודמים. עברו על כל הקבצים וודאו שהשגיאה לא נמצאת בקובץ ששכחתם לבטל בו ניסוי.
* שגיאות שמכילות את הטקסט Ambiguous dependencies או Unsatisfied dependencies אומרות שה Framework פועל, אך ההזרקה לא הוגדרה כראוי.
* בכל פעם שאתם משנים אנוטציה ב Producer, ודאו ששיניתם אותה גם בנקודת ההזרקה.
* בכל פעם שאתם משנים את הפרוייקט ומעוניינים להריץ אותו, בנו אותו מחדש:
  + Build | Rebuild project
* רצוי ליצור את הפרוייקט בתיקיה שהמסלול שלה אינו כולל תווים בעברית

# נספח 2: קוד לדוגמא להתנסות ראשונית עם AOP

תוכלו להשתמש בקוד הבא (שש מחלקות: Pet, Dog, Student, Loggable, Logging ו MainApp) כדי להתנסות ב AOP (אל תשימו את הקוד שלכם ישירות בתיקיה src/java, אלא צרו חבילה חדשה מתחתיה). שימו לב שהפלט הצפוי כולל הודעות שמופיעות באדום המכילות מידע על אתחול המערכת. ריצה תקינה תוביל להדפסות נוספות, פרט להודעות אלה.

**package** student;  
@Retention(***RUNTIME***)

**public** @**interface** Loggable {  
  
}

**package** student;  
**public interface** Pet {  
}

**package** student;  
**public class** Dog **implements** Pet{  
 @Override  
 **public** String toString() {  
 **return "Dog"**;  
 }  
}

**package** student;  
**import jakarta.inject.Inject;**

**public class** Student {  
 **private** Integer **age**;  
 **private** String **name**;  
  
 Pet **itsPet**;  
  
 @Inject  
 **public** Student(Pet pet){  
 **itsPet** = pet;  
 }  
 **public void** setAge(Integer age) {  
 **this**.**age** = age;  
 }  
 **public** Integer getAge() {  
 **return age**;  
 }  
  
 **public void** setName(String name) {  
 **this**.**name** = name;  
 }  
 @Loggable  
 **public** String getName() {  
 **return name**;  
 }  
  
 @Override  
 **public** String toString() {  
 **return "Student, name: "** + **name** + **", age: "** +**age**+**", pet: "** + **itsPet**;  
 }  
}

**package** student;  
**import** org.aspectj.lang.JoinPoint;  
**import** org.aspectj.lang.annotation.\*;  
  
@Aspect  
**public class** Logging {

@Pointcut(**"execution(\* \*.\*.set\*(..))"**)  
 **private void** selectSetters(){}  
  
 @Pointcut(**"@annotation(Loggable) && execution(\* \*(..))"**)  
 **private void** selectLoggable(){}  
  
 @Before(**"selectSetters()"**)  
 **public void** beforeSetters(JoinPoint jp){  
 System.***out***.println(**"Calling "** + jp.getSignature().getName()  
 + **" with "** + jp.getArgs()[0]);  
 }  
  
 @Before(**"selectLoggable()"**)  
 **public void** beforeLoggable(JoinPoint jp){  
 System.***out***.println(**"Calling loggable function "** + jp.getSignature().getName() + **" from "**+jp.getThis());  
 }  
}

**package** student;

import org.jboss.weld.environment.se.Weld;

import org.jboss.weld.environment.se.WeldContainer;  
**public class** MainApp {  
 **private static** WeldContainer *container* = **new** Weld().initialize();

**public static void** main(String[] args) {  
 Student student = container.select(Student.**class**).get();  
 student.setAge(22);  
 student.setName(**"Assaf"**);  
 student.getAge();  
 student.getName();  
 }  
}