**תכנית בסיסית בJAVA**

**שימו** ♥ יכול להיות שחלק מהדברים לא יהיו מובנים לעומק, תקראו אותם ותבינו את הרעיון הכללי, בשיעורים הבאים נכסה את הנושאים בצורה יותר מפורטת. זכרו, המטרה בקישור הנ"ל היא להבין את המבנה הבסיסי של תכנית Java.

**למה צריך שפות תכנות?**   
יצרו שפות תכנות כדי שלנו, בני האדם, יהיה נוח לתקשר עם שפת המחשב, שהיא שונה ומסובכת הרבה יותר משפת קוד המבוססת שפה מדוברת (אנגלית) וידידותית.

המחשב עובד בשפה בינארית, זוהי השפה היחידה שהמחשב באמת "מבין":

שפה בינארית היא שפה הבנויה אך ורק משתי ספרות – 0 ו1.

להרחבה: <https://en.wikipedia.org/wiki/Binary_code>

היות ולנו – בני האדם, **מאוד לא נוח** לכתוב פעולות למחשב בצורה הזו – הומצאו שפות התכנות, שאלו שפות אשר דרכם אנו מגדירים פעולות למחשב, בצורה שמובנת לנו.

מאחורי הקלעים שפות תכנות אלו מומרות לשפה הבינארית – השפה של המחשב. כך הדברים שאנחנו מתכנתים רצים בסוף בפועל.

**תהליך הקומפילציה (הידור) והצורך בו**

<https://www.youtube.com/watch?v=8zmaYXNiwZk>

מושגים חשובים ב-Java שחשוב שנכיר בתור התחלה הם : JVM, JDK, JRE.  
ניתן לקרוא הסבר עליהם ועל ההבדלים שלהם [בלינק הבא](https://www.guru99.com/difference-between-jdk-jre-jvm.html#:~:text=JDK%20is%20a%20software%20development%20kit%20whereas%20JRE%20is%20a,JVM%20is%20Java%20Virtual%20Machine.).

**לפני** שאנחנו יוצרים תכנית בסיסית בJAVA, עלינו לבצע את השלבים הבאים:

* עלינו להוריד ולהתקין JDK על המחשב שלנו.  
  הJDK שנוריד בא ביחד גם עם JRE, וה JRE מכיל בתוכו JVM שהוא חלק ממנו.
* לוודא שאכן הורדנו והתקנו את [סביבת העבודה](https://he.wikipedia.org/wiki/%D7%A1%D7%91%D7%99%D7%91%D7%AA_%D7%A4%D7%99%D7%AA%D7%95%D7%97_%D7%9E%D7%A9%D7%95%D7%9C%D7%91%D7%AA) intellij.  
  במידה ולא יש לחזור לשלב הקודם ולעשות זאת!

לאחר שלמדנו מעט על שפת ותהליך הקומפילציה, הגיע הזמן לכתוב תכנית בסיסית שנוכל לבסוף להריץ ולראות אותה בפעולה. לפני שאנחנו מתחילים במימוש של תכנית גדולה בJava, עלינו להבין את הבסיס המעשי.

**מבנה ומונחים בסיסיים**

תכנית בשפת Java מורכבת מאוסף של פקודות קוד.  
בזמן ריצת התוכנית, הפקודות מופעלות זה אחר זה לפי סדר כרונולוגי מלמעלה למטה.  
כל פקודה מורה למחשב לבצע פעולה פשוטה מסוימת ו\או אוסף של פעולות פשוטות, שלבסוף יכולות לייצר תכנית שמבצעת תהליך מורכב.

לפני שנתחיל עלינו להכיר את המונחים הבאים:

* **הערות -** קטעי טקסט אשר ניתן להוסיף לקוד שלנו במטרה לסייע למפתחים אחרים בהבנה של מהלך הקוד ומטרתו.  
  בזמן הריצה, התוכנית פוסחת על קטעי הטקסט האלה, והם לא משפיעים כלל על הריצה של התוכנית.
* **Import -** לפעמים אנו נרצה לייבא ספריות חיצוניות המכילות פונקציונאליות נוספת. (ספריה מכילה קוד או מידע המספק שירות מסוים לצורך הפיתוח שלנו)   
  נעשה זאת ע"י שימוש בפקודת import.
* **Main –** "אזור" בו נכתוב את הקטע המרכזי של הקוד. במהלך הקורס נלמד כיצד ניתן לחלק את הקוד ל"אזורים" שונים ונשתמש ב"מחלקות" ודברים נוספים. נכון לעכשיו נשתמש ב-main כאזור קטע הקוד המרכזי שלנו בו נכתוב את כל הקוד. ה-main יהיה תחום בסוגריים מסולסלים { } אשר נקראים ביחד "בלוק", המייצגים את תחילתו ואת סופו של קטע זה.

לפני שאתם ממשיכים בנספח, עליכם לקרוא עוד על מבנה תכנית בסיסית בJava בלינק הבא, אשר מסביר את כל הנושאים יותר לעומק.

<https://www.w3schools.in/java-tutorial/program-structure/>

**חשוב לדעת!**

כל עוד מותקן לנו על המחשב JDK, אנחנו יכולים ממש להתחיל לפתח ולתכנת בJAVA – ניתן לכתוב קוד של JAVA ממש בnotepad.

ניתן ליצור תיקייה חדשה הכוללת את הקבצים הדרושים בצורה מסוימת, ולאחר מכן להעביר אותם תהליך קומפילציה באמצעות שימוש בCmd (Command line interface) ולהריץ את פרויקט הJava.

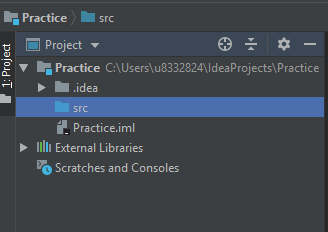
סביבת העבודה intellij למעשה מאפשרת לנו לפתח קוד בJAVA **בצורה נוחה יותר**.  
לכן, בתור מפתחים תמיד נעדיף לפתח ולכתוב את הקוד שלנו באמצעות סביבת עבודה (כאמור intellij במקרה שלנו).

כעת, ניצור תכנית JAVA בסיסית לפי ההגדרות הנפוצות:

* נבחר בFile->New->Project

אנו יכולים לתת לפרויקט שלנו איזה שם שנבחר, בדוגמה הזו נקרא לפרויקט “**Practice**”

בצד שמאל ניתן לראות את עץ התיקיות של הפרויקט שלנו:



אין צורך להבין את כל המבנה תיקיות והקבצים בשלב זה, נעמיק ונלמד אותו בהמשך.

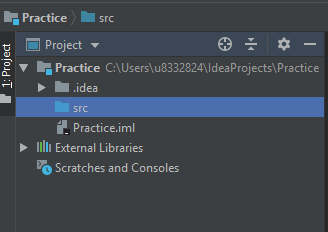
כעת, לאחר שיצרנו פרויקט, השלב הראשון שעלינו לנקוט בו הוא **יצירת מחלקה.**

[מחלקה](https://he.wikipedia.org/wiki/%D7%9E%D7%97%D7%9C%D7%A7%D7%94_(%D7%AA%D7%9B%D7%A0%D7%95%D7%AA)) היא ישות כללית בתכנות מונחה עצמים בכלל, ובשפת Java בפרט, אשר מכיל בתוכה את הישויות הבסיסיות 'משתנים' , 'פונקציות' וכו' - גם על מושגים אלו נרחיב בהמשך.

עליכם ליצור מחלקה אחת ולקרוא לה "Main".

(ניתן לקרוא לה בכל שם אחר לצורך העניין, אך אנחנו נקרא לה **Main** כסטנדרט).

* נרחיב את תיקיית הפרויקט שלנו, אמורה להופיע תיקייה בשם "src".



* Graphical user interface, text, application

  Description automatically generatedנעמיד את העכבר מעל תיקיית **ה - src**, נלחץ על המקש הימני ->New -> Java Class.
* ניצור מחלקה חדשה בשם "Main".

כעת נוצר לנו **קובץ** Main.java אשר נמצא בתיקיית src.

אבל! עדיין לא סיימנו...

אומנם יש לנו מחלקה, אבל אנחנו צריכים להוסיף בתוך המחלקה הזו את ה"אזור" בו נכתוב את הקטע המרכזי של הקוד – **הMain.**

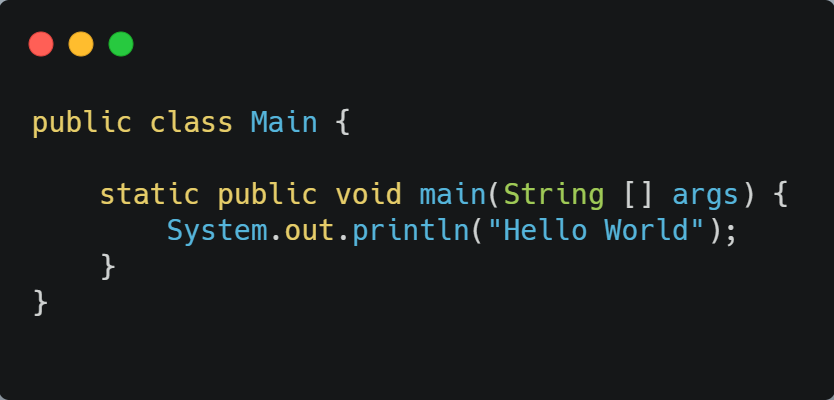
נכתוב בתוך המחלקה את הקטע קוד הבא, שמצהיר על יצירת הMain:

static public void main(String [] args) {}

הסוגריים המסולסלים { } בסוף השורה נקראים "בלוק". הסוגריים מייצגים את תחילתו ואת סופו של ה"בלוק" ובתוכו נרשום את הקוד שלנו.

בתוך הבלוק נכתוב את הקוד הבא ה'מדפיס' הודעה למשתמש:   
**(נעמיק על פקודת ההדפסה בהמשך הקורס הרשתי, כרגע רק תכירו איך היא נראית)**

System.*out*.println("Hello World");



**ניתן לראות ולערוך את הקוד**   
[כאן](https://carbon.now.sh/?bg=rgba(171%2C%20184%2C%20195%2C%201)&t=seti&wt=none&l=auto&ds=true&dsyoff=20px&dsblur=68px&wc=true&wa=true&pv=0px&ph=0px&ln=false&fl=1&fm=Hack&fs=14px&lh=133%25&si=false&es=2x&wm=false&code=public%2520class%2520Main%2520%257B%250A%2520%2520%2520%2520%250A%2520%2520%2520%2520static%2520public%2520void%2520main(String%2520%255B%255D%2520args)%2520%257B%250A%2520%2520%2520%2520%2520%2520%2520%2520System.out.println(%2522Hello%2520World%2522)%253B%250A%2520%2520%2520%2520%257D%250A%257D%250A)

- האם ניתן ליצור מחלקה שמכילה main בצורה אוטומטית בלי שנצטרך לכתוב את כל הקוד הזה בעצמנו? **בדקו באינטרנט!**

**כיצד "נריץ" את הקוד?**

* נלחץ מקש ימני על שם המחלקה שלנו
* נלחץ על הכפתור 'play' הירוק – run Main.main
* כעת נראה למטה את ההדפסה Hello World !

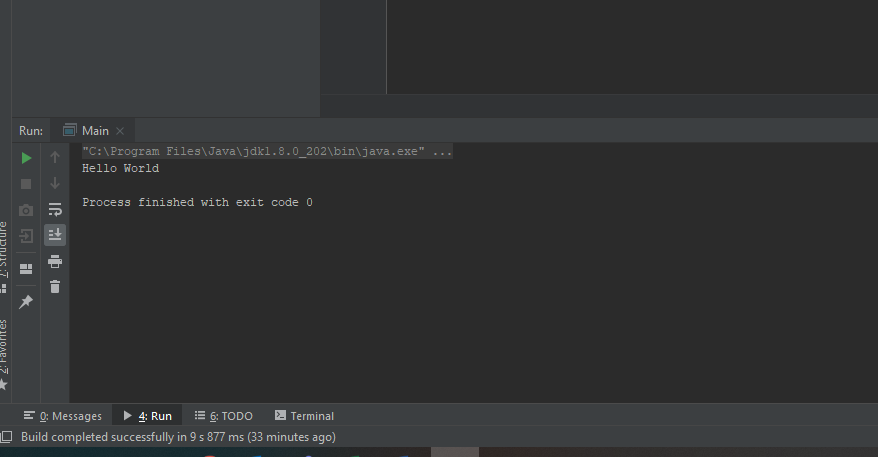
- האם יש קיצור מקלדת שבו ניתן להריץ את הקוד בצורה נוחה יותר?   
**בדקו באינטרנט!**

**כתיבת הערות**  
לעתים הקוד שנכתוב יהיה מורכב ולא בהכרח מובן למפתחים אחרים שיצטרכו להבין את הקוד שכתבנו לאור שינויים עתידיים. בעקבות כך, מאפשרים בשפות תכנות **לכתוב** הערות בקבצי הקוד שלנו, ממש בין השורות.

הערות בקוד הן למעשה טקסט שאנחנו כותבים ביחד עם הקוד שלנו (במקרה שלנו בקובץ Main.java) אבל הקומפיילר יודע להתעלם ממנו.

* קראו באינטרנט כיצד לכתוב הערות בJava.
* כתבו הערות בפרויקט שיצרתם.

**אז מה עשינו עד כה?**

* יצרנו פרויקט Java באמצעות סביבת העבודה intellij המותקנת על המחשב שלנו.
* יצרנו **מחלקה** אשר מהווה את הקובץ הבסיסי שבתוכו נמצא הקוד שלנו.
* יצרנו את איזור ה - main שמכיל את הקוד בפרויקט שלנו.  
  בהמשך הקורס נלמד כיצד ניתן ליצור כמה חלקים שונים בקוד ולהיעזר בהם, כרגע נכתוב את כל הקוד בתוך ה-main.
* הוספנו פקודת אשר **מדפיסה** למסךהמשתמש.  
  הפקודה הזו למעשה מדפיסה את הפלט ב-console של סביבת העבודה שלנו, ניתן לראות את הפלט ב-console למטה לאחר הרצת הקוד.

בהמשך נלמד לכתוב פרויקטים מורכבים המשלבים לוגיקה מורכבת, עם המון פונקציות, קבצים וספריות שונות, וע"י כך ליצור פרויקטים מעניינים.

בהמשך הקורס, השיעורים הבאים יהיו מאתגרים ולכן חשוב מאוד להבין את הבסיס ולהכיר את כל הטריקים וקיצורי הדרך שיש לסביבת העבודה שלנו להציע בשפת תכנות Java.

**תרגול**

1. צרו פרויקט JAVA **חדש** המדפיס את הנתונים הבאים:  
   שם מלא, גיל, וחיית מחמד אהובה.

בהצלחה !