Tutorial de instalare şi utilizare a mediului Eclipse

2016

1 Scopul lucrării

Obiectivele acestei sesiuni de laborator sunt:

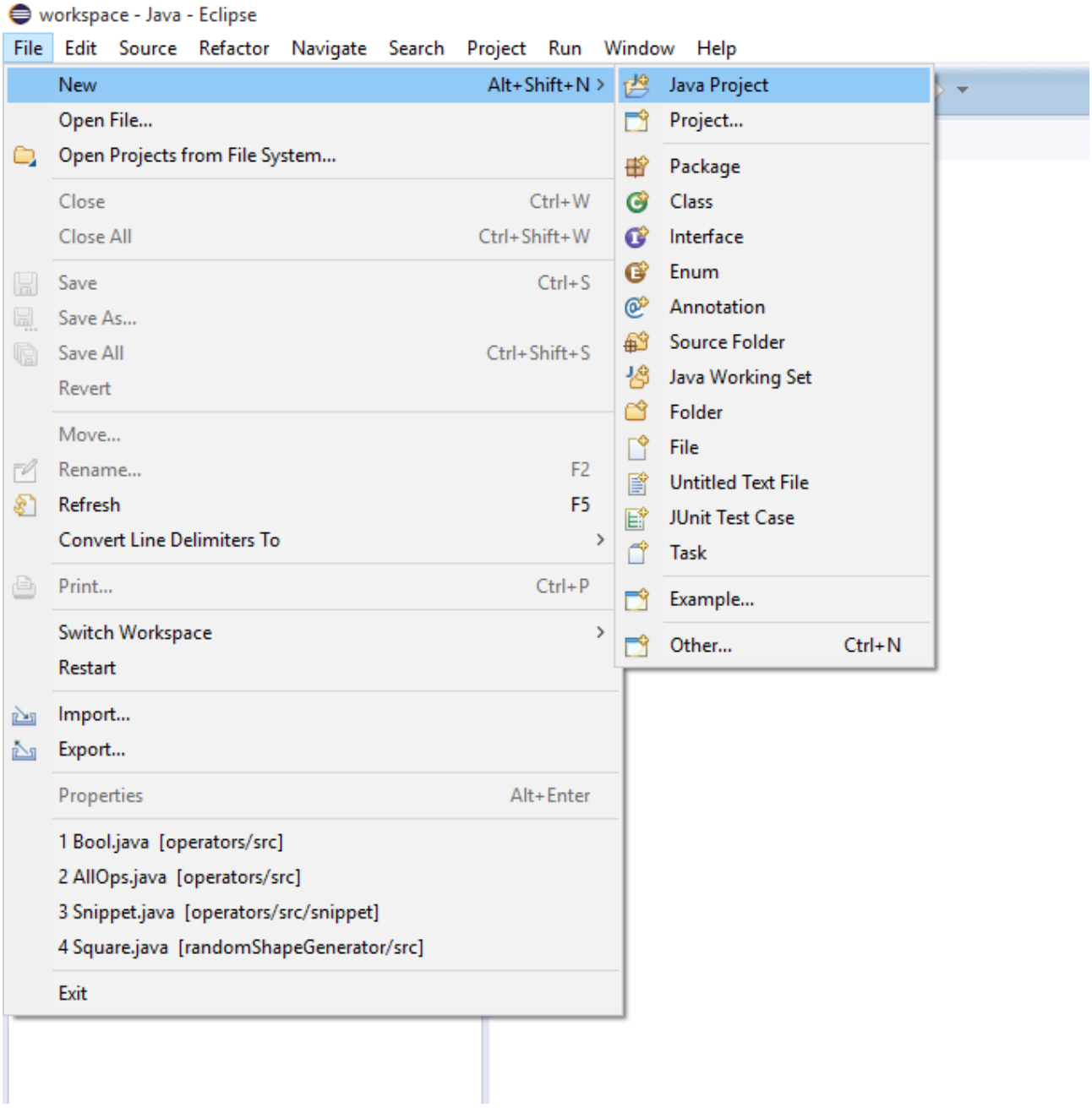
- Instalarea şi familiarizarea cu mediul de lucru Eclipse.
- Crearea câtorva programe java simple.

2 Instalare Eclipse

1. Descãrcaţi fişierul de instalare .exe de pe site-ul: https://eclipse.org/downloads/
2. Rulaţi fişierul .exe cu drept de administrator şi urmaţi paşii necesari pentru instalare.

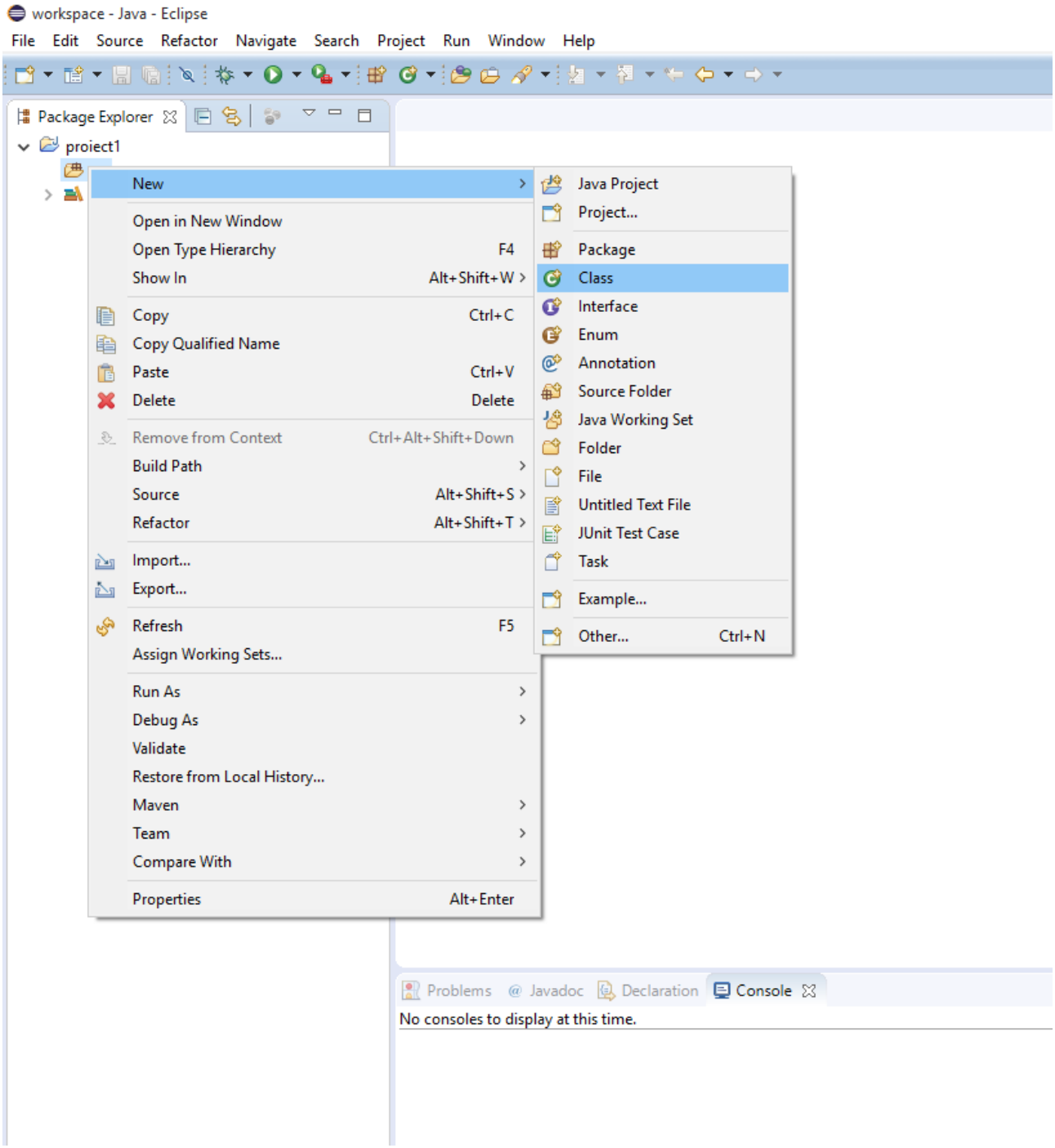
3 Crearea unei aplicaţii “Hello world” simple

3.1 Creaţi un proiect nou din meniul File → New → Java Project

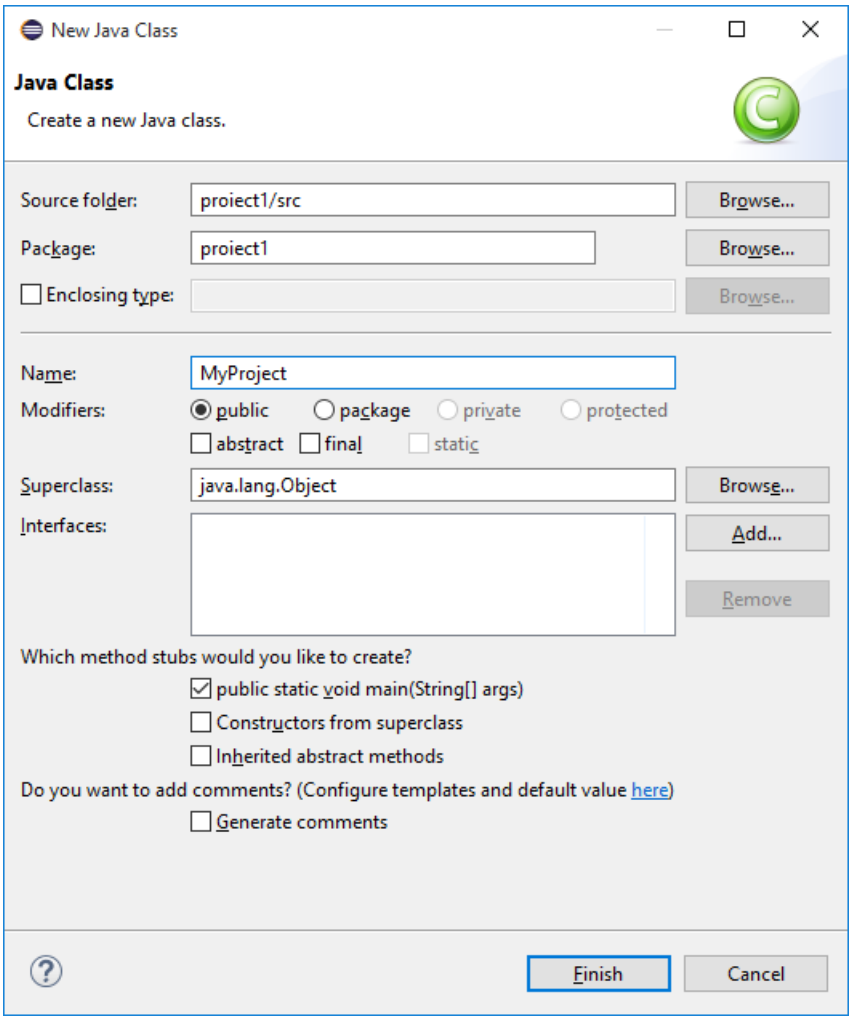


3.2 Dati un nume proiectului, selectaţi fişierul unde vreţi sã-l salvaţi pe disc, apoi apãsaţi butonul „Finish”.

3.3 În proiectul creat, daţi click dreapta pe pachetul src creat automat şi creaţi o clasã nouã, ca în figura de mai jos:



3.4 Daţi un nume clasei pe care o creaţi (numele claselor vor începe întotdeauna cu majusculă!). Puteţi selecta opţiunea de a crea automat funcţia main - ce va fi executată la rularea programului:



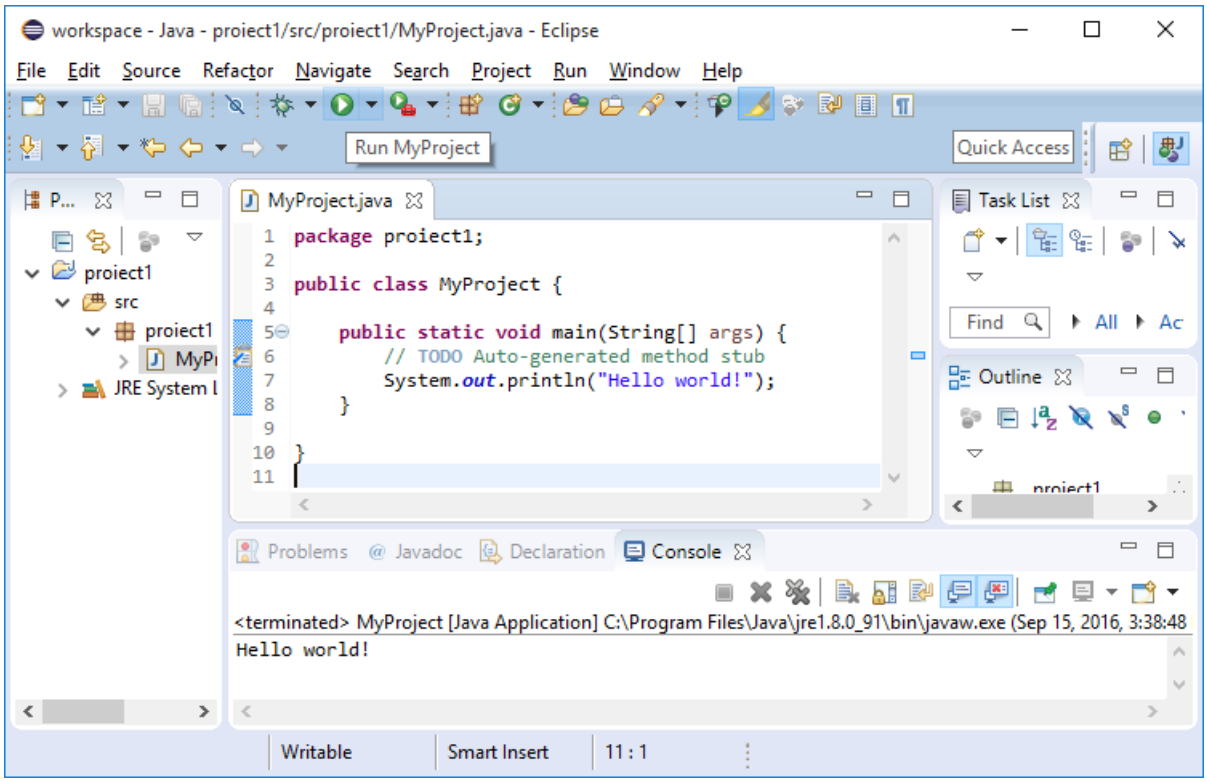
3.5 La apăsarea butonului Finish se va crea un fişier MyProject.java cu următorul cod:

```
1 public class MyProject {
2     public static void main(String[] args) {
3         // TODO Auto-generated method stub
4     }
5 }
```

3.6 Vom scrie codul ce dorim să se execute în funcţia main. De exemplu, pt. afişarea mesajului "Hello world", vom scrie următoarea linie de cod:

```
1 System.out.println("Hello world!");}
```

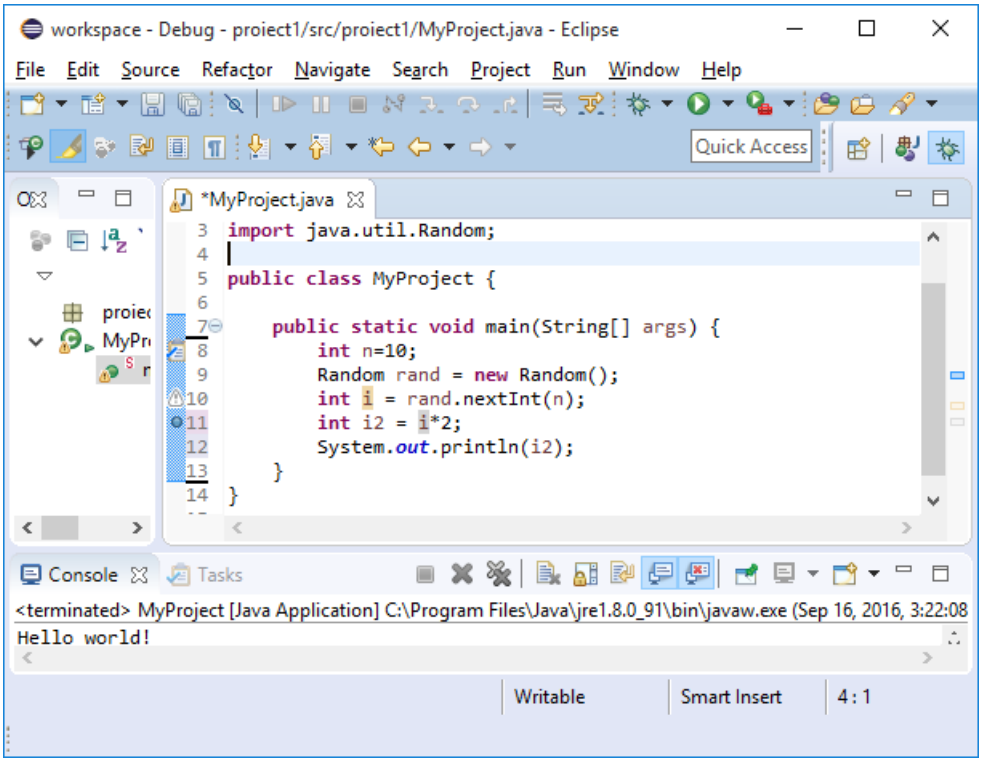
3.7 La rulare, outputul se afisează în consolă, ca în figura alaturată



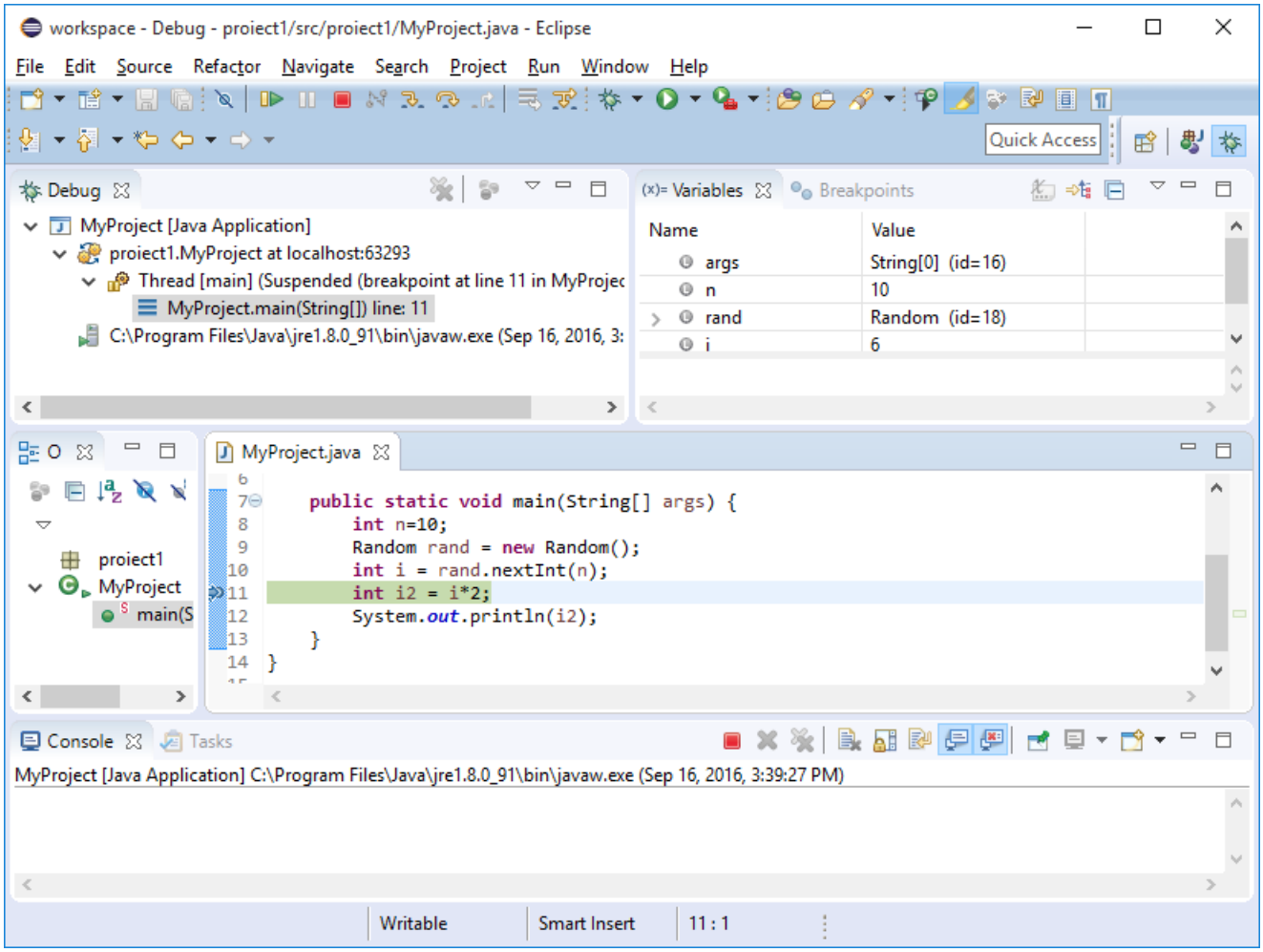
4 Folosirea debugger-ului

Pentru a depana un program şi a găsi mai uşor greşelile, este recomandată folosirea debuggului. Depanarea presupune poziţionarea de break point-uri în dreptul liniilor unde vrem să oprim execuţia programului pentru a putea verifica valabilitatea valorilor din variabilele curente.

4.1 Activarea unui break point în dreptul liniei unde vrem să oprim execuţia programului se face dând dublu click în dreptul acelei linii de cod, iar apoi rularea se face in modul debug, ca în figura de mai jos:



4.2 La rulare, execuţia se opreşte În dreptul liniei unde am plasat break point-ul, arãtând astfel:



Fereastra Variables se deschide automat. Aici puteți vizualiza valorile variabilelor locale.

4.3 În continuare aveți posibilitatea

- de a continua rularea programului (Resume F8);
- de a merge la urmãtoarea linie de cod care se executã (Step Into F5). Dacã pe linia curentã este apelul unei funţii, atunci opţiunea Step Into va intra în interiorul funcţiei;
- de a merge la urmãtoarea linie de cod din funcţia curentã (Step Over F6).

5 Mersul lucrării

5.1 Creati o aplicaţie simplă care să afişeze mesajul "Hello world" urmând paşii prezentaţi mai sus.

5.2 Adăugaţi o funcţie care să returneze valoarea maximă dintre două numere.

5.3 Adăugaţi o funcţie care are ca şi parametru un număr întreg şi returnează dacă numărul este prim sau nu.

5.4 Scrieţi un program care să simuleze jocul de cărţi 21. Jocul are doi jucători care vor întoarce pe rând cărti de joc cu valori între 1 si 11. Când suma carţilor se apropie de valoarea 21, jucătorii pot decide să se oprească. Câştigă jucătorul cu suma cea mai mare care nu depăşeşte valoarea 21.

6 Sugestii implementare

- Generarea unui număr aleator Între 0 si n se face cu ajutorul funcţiei presefinite Random astfel:

```
Random rand = new Random();
int i = rand.nextInt(n);
```

- Pentru a stabili dacă jucătorul doreste sa continue, se poate citi o codificare 1, sau 0 din conolă. Citirea unui număr din consolă se poate face folosind clasa predefinită Scanner. ex:

```
Scanner in = new Scanner(System.in);
int continua = in.nextInt();
```