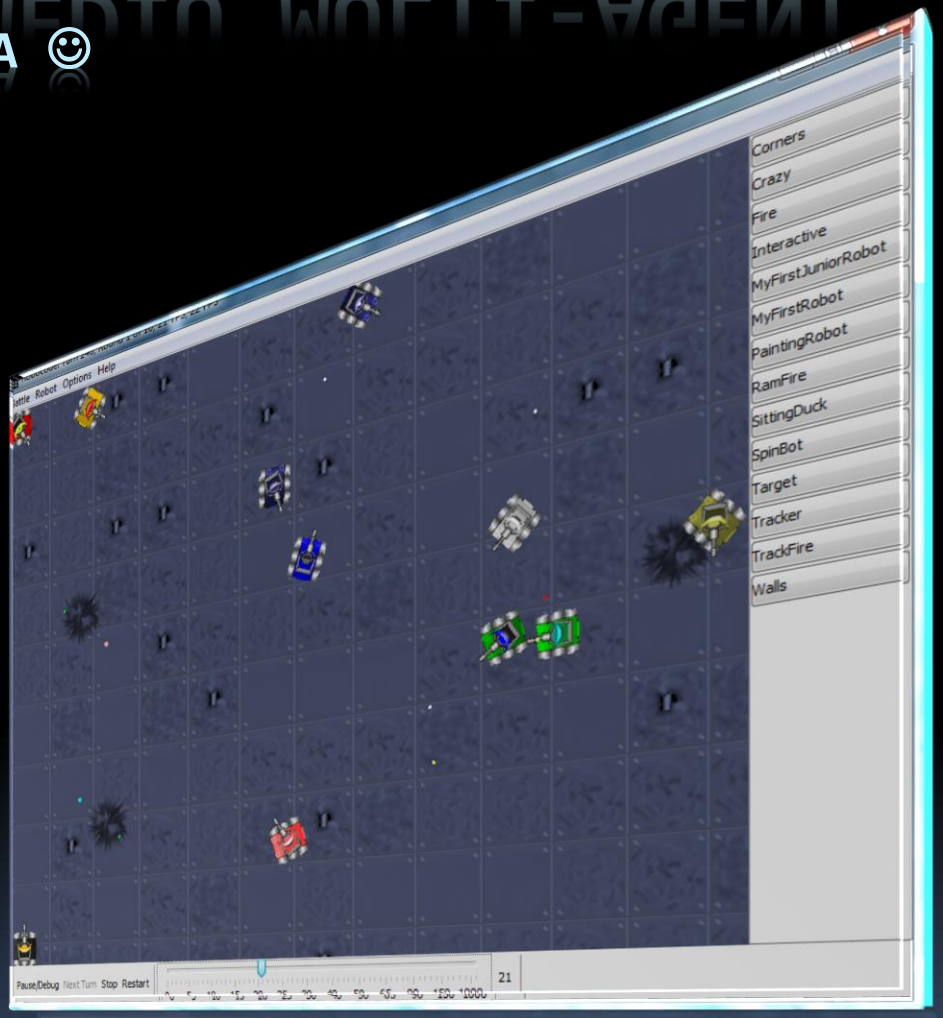
ROBOCODE – UN MEDIU MULTI-AGENT

SI O SANSA SA INVATATI JAVA ☺

Catalin Stoean



Introducere

- Robocode este o platforma multiagent in care este simulata o lupta.
  - A fost dezvoltata de catre Mathiew Nelson de la IBM in 2001.
- Este scris in Java
  - Initial s-a dorit a fi o unealta de invatare a Java
  - A devenit rapid un joc la nivel mondial
    - si o unealta de cercetare a inteligentei artificiale
- Creezi un robot printr-un program Java , il arunci in campul de bataie unde lupta pana la sfarsit cu alti roboti creati de alti programatori.





Scopul acestui curs

- Creati-va propriul robot.
  - Dati-i miscari mecanice… apoi…
  - Oferiti-i inteligenta artificiala!
- Robotul poate fi ghidat de:
  - Algoritmul minimax
  - Invatare reimprospatata
- Aduceti-va robotii pe acelasi camp de lupta!





Instalarea Robocode

- Inainte de a instala Robocode, aveti nevoie de Java (JDK) :

  http://java.sun.com/javase/downloads/index.jsp

- Pentru Robocode, descarcati de la pagina :

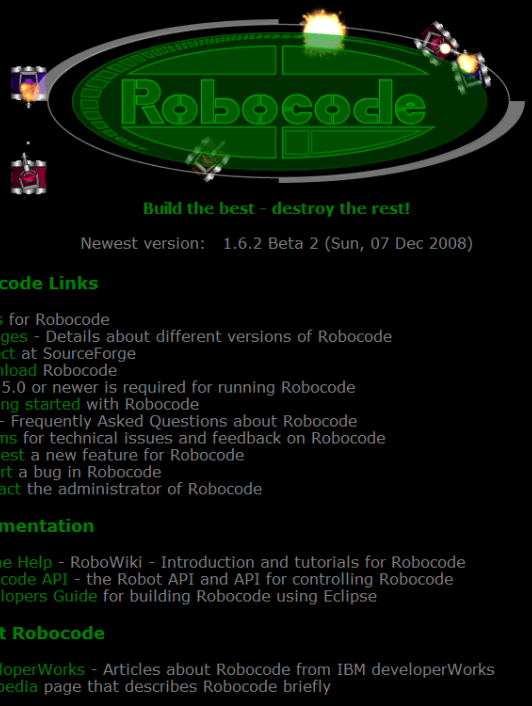
http://sourceforge.net/projects/robocode/files/robocode/1.9.2.4/

sau unul mai recent daca exista.

- Daca ati instalat JDK, puteti da dublu-click pe

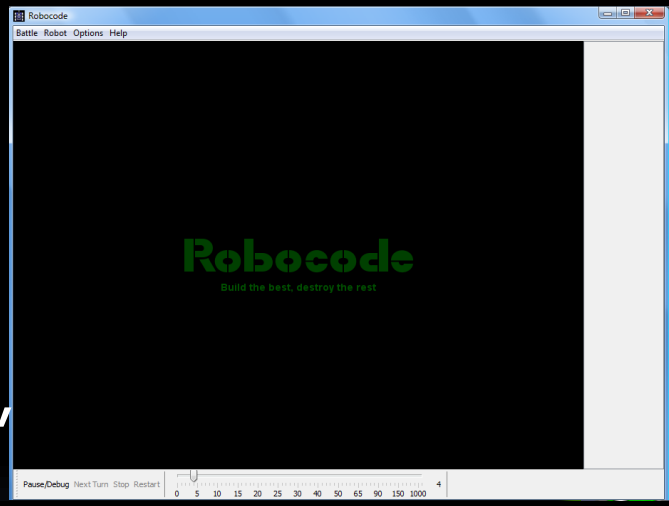
robocode-1.9.2.4-setup.jar

si se va face instalarea.

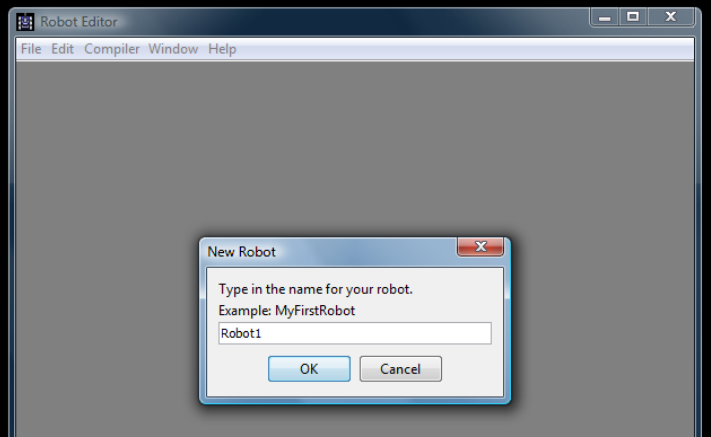


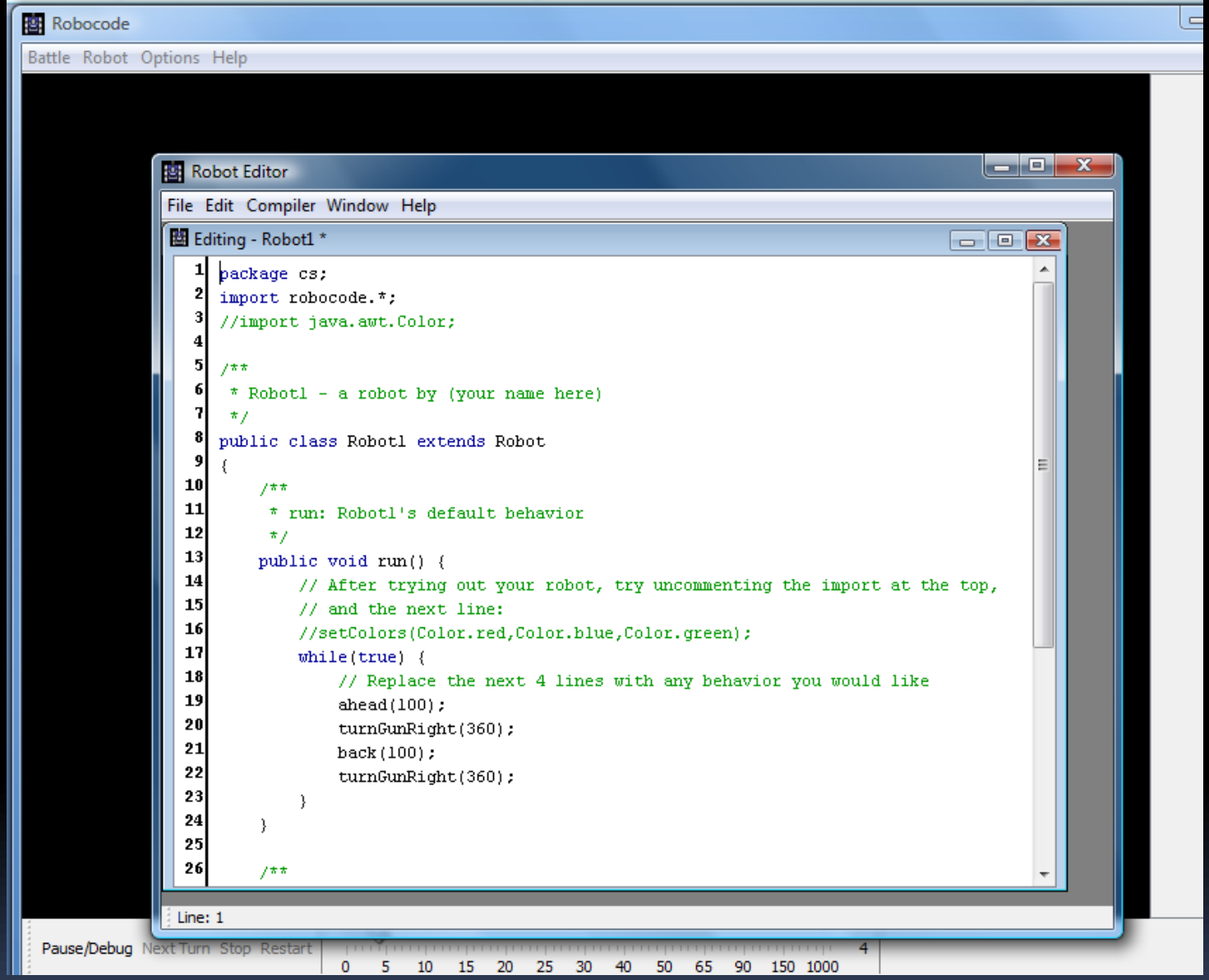
Lansarea Robocode

- Lansati Robocode de la shortcut-ul de pe Desktop.
- Pentru a crea un robot, selectati din meniu “Robot”, apoi “Editor”. Se deschide o fereastra, unde mergeti la meniul “File”->”New”->”Robot”



- Ii puneti un nume, apoi va va cere sa ii dati initialele (la mine, “cs”.)

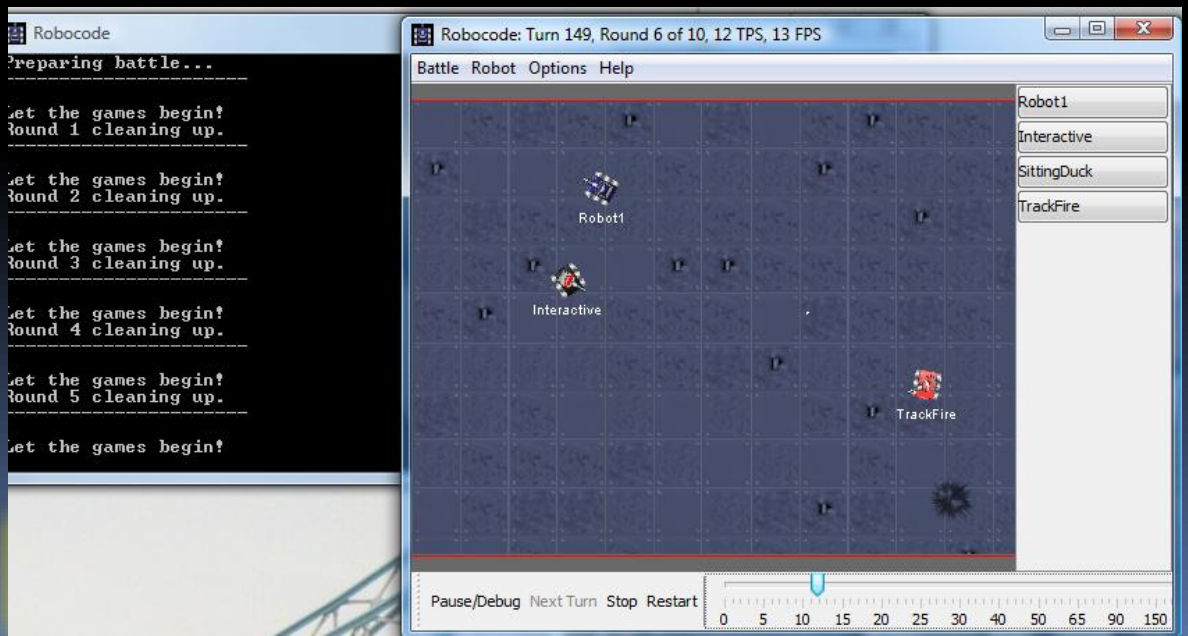




- Nu uitati sa salvati codul (din meniul File) si sa il compilati (din meniul Compiler)!

Cum ajungem in arena?

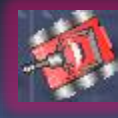
- Selectati meniul *Battle*.
- Alegeti robotii pe care vreti sa ii antrenati intr-o batalie (inclusiv al vostru).
- Daca robotul vostru a supravietuit bataliei, ati castigat *runda*.



Ce poate face un robot

- Robotii simuleaza tancuri intr-o arena de lupta si, pentru a detecta alti roboti, sunt echipati cu radare.
- Un robot poate merge inainte si inapoi cu diverse viteze si se poate intoarce la stanga si la dreapta.
- Radarul si tureta se pot intoarce la stanga sau dreapta, fiecare independent si, in acelasi timp, si de restul tancului. Arma poate, bineinteles, trage.
- Cand un robot inamic apare pe radar, un eveniment Java este generat si se pot genera diverse actiuni.

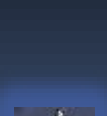




Ce informatii se pot obtine

- Putem lua informatii despre robotul inamic detectat in ceea ce priveste:
  - Viteza sa
  - Modul in care este indreptat
  - Energia ramasa
  - Numele
  - Unghiul dintre orientarea propriului robot si robotul inamic detectat
  - Distanta pana la acel robot





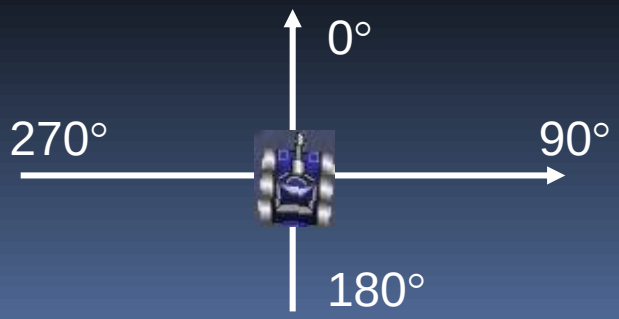
Campul de lupta

- Este reprezentat intr-un plan bidimensional si este imprejmuit de ziduri.
- O pozitie din campul de lupta este data de coordonatele (x, y)
  - Originea (0, 0) este plasata in coltul din stanga jos.
- Pentru a lua informatii despre marimea campului de lupta si despre pozitia robotilor, se pot folosi urmatoarele metode ale ale clasei **Robot**:

- getBattleFieldHeight() - getX()
- getBattleFieldWidth() - getY()

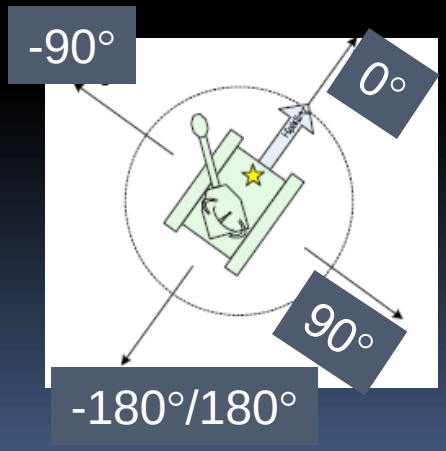
Orientarea si manevrarea

- Cand un robot se misca, el are o orientare.
  - Orientarea poate fi obtinuta prin apelarea metodei “getHeading()” din cadrul clasei **Robot**.
  - Se poate obtine orientarea propriului robot, dar si cea a altor roboti care au fost detectati cu radarul.
- Orientarea este masurata in sensul acelor de ceasornic de la 0° la 360°.
  - Orientarea de 0° se obtine cand tancul este cu fata la nord.

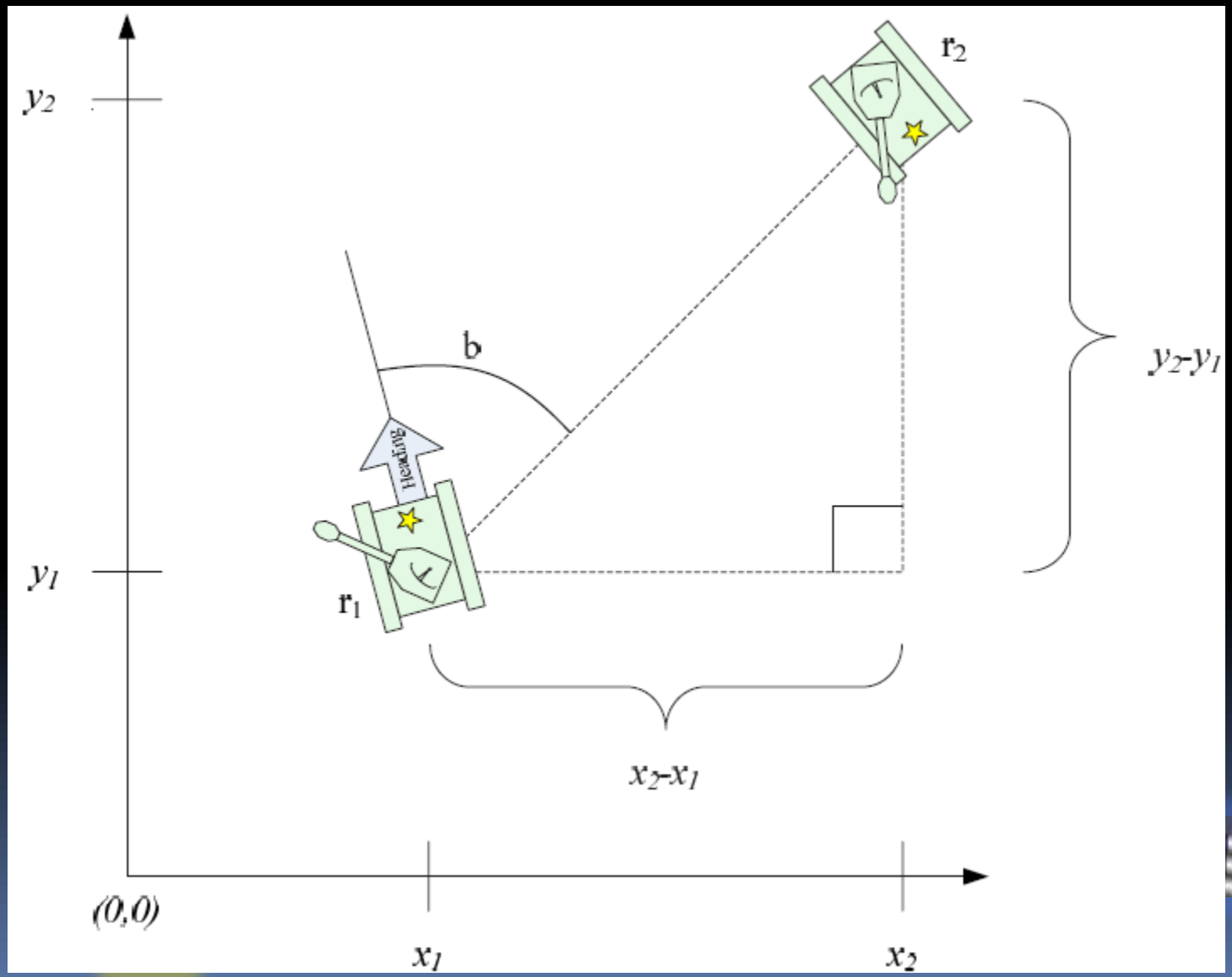


Orientarea si manevrarea

- Manevrarea se poate face in intervalul [-180°, 180°].
- O manevrare de la 0° la -180° va determina o mutare stanga a robotului, iar intre 0° si 180° una la dreapta.



Orientarea si manevrarea



Puterea si energia

- Distanta se masoara in pixeli. Cel mai mic camp de lupta are 400x400 pixeli, iar cel mai mare are 5000x5000.
- Fiecare robot are o energie de 100 la inceputul jocului
  - Pierde energie in timpul jocului daca este lovit.
  - Castiga daca loveste alti roboti.
- Puterea este cantitatea de energie pe care o pune intr-un explozibil cand trage.
  - Cu cat lovitura este mai puternica, cu atat se consuma mai multa energie.





Puterea si energia

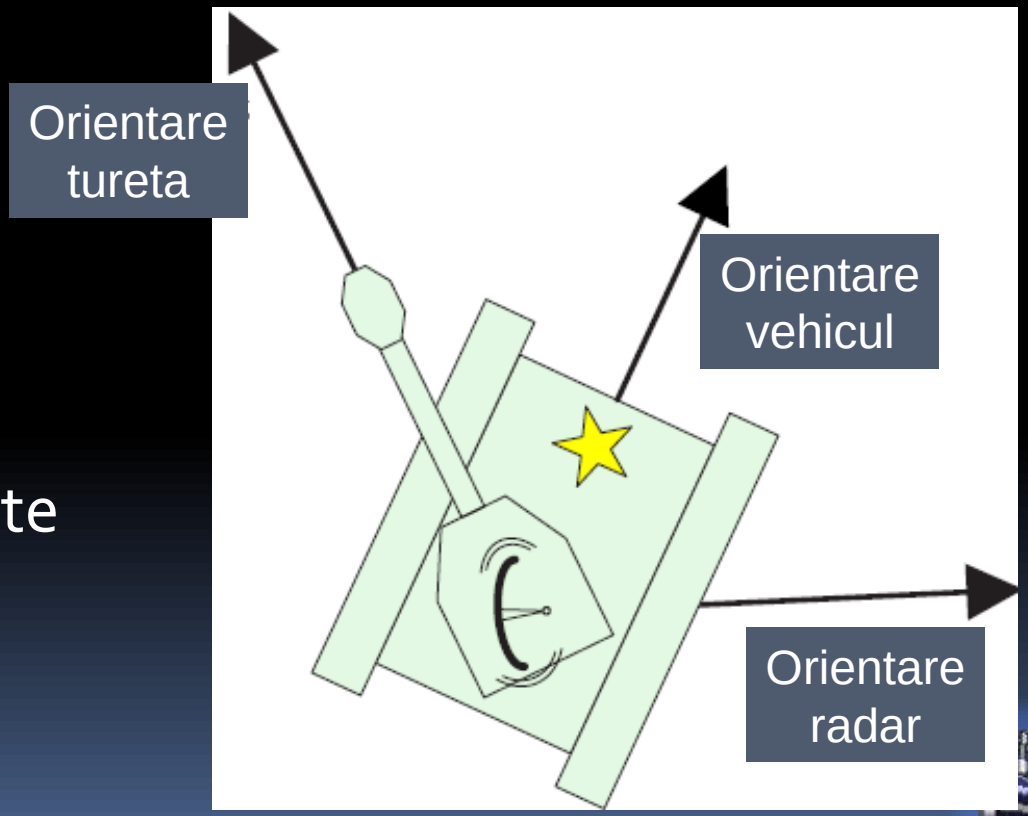
- Daca un robot trage cu o putere de 2.5 in alt robot, din energie se scade exact 2.5.
- Puterea poate lua valori in intervalul [0.1, 3]
- Daca lovitura noastra (de putere p) loveste un robot inamic, energia creste cu 3p.
- Ca sa vedem cata energie avem la momentul curent, apelam metoda getEnergy() din clasa Robot.
- Atunci cand robotul nostru este lovit, energia sa scade dupa urmatoarea formula:
  - energie = energie – 4p, daca $p \leq 1$
  - energie = energie – 4p – 2 (p-1), daca $1 < p \leq 3$
- Cand ne lovim de un alt robot, energia pierduta este de 0.6.
- Cand ne lovim de zid:
- energie = energie – $|v|/2 + 1$, unde v este viteza.





Alcatuire robot

- Fiecare robot are 3 componente:
  - Un radar
  - O tureta
  - Vehicul
- Cele 3 parti se pot misca independent, dar sunt montate una deasupra celeilalte.



Timp, viteza

- Timpul se masoara in frame-uri.
- Rata maxima de rotatie a armei (turetei) este de 20°/frame, a radarului de 45°/frame.
- Radarul este montat pe arma, iar aceasta este montata pe vehicul.
  - Daca se misca arma, se misca si radarul, iar daca se misca vehiculul, se misca si arma cu radarul.
  - Ex: daca intoarcem arma la stanga si radarul la dreapta, acesta din urma se intoarce doar cu 45°-20°=25° la dreapta. Pe de alta parte, daca le intoarcem la dreapta pe amandoua, radarul se va intoarce cu 45°+20°=65°.





Timp, viteza

- Viteza unui robot se obtine apeland metoda getVelocity() pentru o instanta a clasei **Robot**.
  - Viteza poate fi doar in intervalul [-8, 8] pixeli/frame.
- Rata la care se poate intoarce un vehicul depinde de viteza sa.
  - Avand o viteza v, rata de intoarcere (masurata in grade/frame) este

$$t(v) = 10 - |v| * 3/4$$





Alte informatii utile

- Dupa foc, tureta se incinge si nu poate trage un alt foc pana cand nu se raceste.
  - Cu cat puterea focului este mai mare, cu atat arma se incalzeste mai tare
  - Inainte de a trage un foc, interogati daca arma este fierbinte, ca sa fiti siguri de reusita.
- Radarul nu detecteaza explozibilul adversarilor.





Clase

- Clasele pe care le poate mosteni un robot creat de noi sunt **Robot** si **AdvancedRobot**.
  - Principalul avantaj al celei de a doua clase fata de prima este ca permite rularea de mai multe procese in paralel:
    - Mutarea vehiculului
    - Mutarea radarului
    - Mutarea turetei

} Concomitent!





Crearea unui robot

```
package cs;
import robocode.*;
/*****     Robot1 - un robot al lui cs          *****/
public class Robot1 extends Robot
{
    //Aici se declara variabile
    /*****                  run() defineste comportamentul robotului Robot1   *****/
    public void run()
    {
        //Cod ce se executa la inceput si o singura data
        while(true)
        {
            //Bucla infinita care determina comportamentul robotului
        }
    }
    //<Aici se scriu metode care spun ce sa faca robotul cand se intampla diverse evenimente
    public void onScannedRobot(ScannedRobotEvent e)
    {
        fire(1);
    }
}
```





Crearea unui robot

Robot Editor

File Edit Compiler Window Help

Editing - C:\robocode\robots\cs\Robot1.java

```
1  package cs;
2  import robocode.*;
3  import java.awt.Color;
4  import java.awt.event.MouseEvent;
5
6  /**
7   * Robot1 - a robot by cs
8   */
9  public class Robot1 extends Robot
10 {
11     /**
12      * run: Robot1's default behavior
13      */
14     public void run() {
15         setColors(Color.red,Color.blue,Color.green);
16         System.out.println("Incepe batalia!!");
17         while(true) {
18             // Replace the next 4 lines with any behavior you would like
19             ahead(100);
20             turnGunRight(360);
21             back(100);
22             turnGunRight(360);
23         }
24     }
25
26     /**
27      * onScannedRobot: What to do when you see another robot
28      */
29     public void onScannedRobot(ScannedRobotEvent e) {
30         fire(1);
31     }
32
```

Crearea unui robot

Robot Editor

File Edit Compiler Window Help

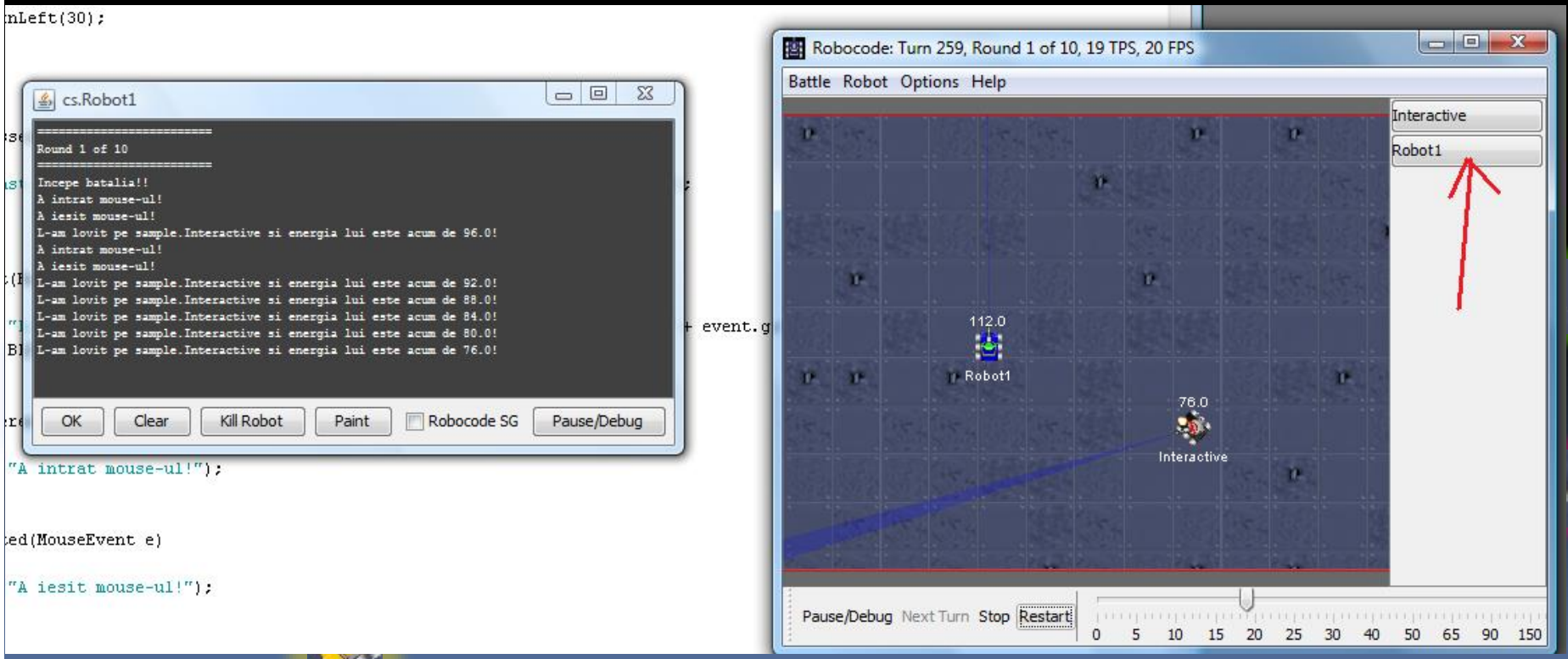
Editing - C:\robocode\robots\cs\Robot1.java

```
32      }
33
34      /**
35       * onHitByBullet: What to do when you're hit by a bullet
36       */
37      public void onHitByBullet(HitByBulletEvent e) {
38          turnRight(e.getBearing());
39      }
40
41      public void onWin(WinEvent e)
42      {
43          out.println("Te-am batut!");
44          for (int i = 0; i < 50; i++)
45          {
46                  setBodyColor(Color.yellow);
47                          fire(0.1);
48                          turnRight(30);
49                          turnLeft(30);
50          }
51
52      }
53
54      public void onBulletMissed(BulletMissedEvent event)
55      {
56          out.println("Am trast la intamplare, iar energia mea este acum de " + getEnergy() + "!
57
58      }
59
60      public void onBulletHit(BulletHitEvent event)
61      {
62          System.out.println("L-am lovit pe " + event.getName() + " si energia lui este acum de
63          setBodyColor(Color.BLUE);
64      }
65
66      public void onMouseEntered(MouseEvent e)
67      {
68          System.out.println("A intrat mouse-ul!");
69      }
70
71      public void onMouseExited(MouseEvent e)
72      {
73          System.out.println("A iesit mouse-ul!");
74      }
}
```




Robotul nostru la lupta

- Pentru lansarea consolei, se apasa butonul indicat mai jos.



Lista de clase si metode folosite

- Din meniul Robocode, mergeti la "Help"->"Robocode API".
- In browserul care se deschide, vedeti in special clasa "Robot" si metodele sale.
- Pentru a accesa API-ul Java, puteti accesa:

  http://docs.oracle.com/javase/7/docs/api/





Lista de clase si metode folosite

- void **ahead**(double distance)
  - Muta robotul inainte cu "distance" pixeli
- void **back**(double distance) // inapoi
- void **fire**(double power)
  - Trage cu explozibil cu puterea power din [0.1, 3.0]
- double **getEnergy**()
  - Intoarce energia curenta a robotului
- double **getGunHeading**()
  - Intoarce directia in care este indreptata tureta, in grade
- double **getHeading**()
  - Intoarce directia vehiculuilui, in grade
- double **getRadarHeading**() //acelasi lucru pt radar
- double **getX**() // pozitia x a robotului
- double **getY**()





Lista de clase si metode folosite

- void **onBulletHit**(BulletHitEvent event)
  - Metoda este apelata cand explozibilul loveste un tanc
- void **onHitByBullet**(HitByBulletEvent event)
  - Metoda este apelata cand robotul este lovit de explozibil
- void **onHitWall**(HitWallEvent event)
  - Cand se loveste de zid
- void **onScannedRobot**(ScannedRobotEvent event)
  - Cand radarul scaneaza un robot
- void **turnGunLeft**(double degrees)
  - Intoarce arma la stanga cu *degrees* grade.
- void **turnLeft**(double degrees)
  - Intoarce vehiculul la stanga cu *degrees* grade.
- void **turnRadarLeft**(double degrees) // radarul





Editor… Eclipse?

- Editorul pus la dispozitie de Robocode este bun, poate si compila, dar mult mai util este Eclipse sau NetBeans.
  - Ambele pot fi descarcate de pe Internet.
  - Eu prefer Eclipse. ☺
  - Cautati "Eclipse IDE for Java Developers "
- De ce sunt mai bune? Printre altele…
  - Compileaza in timp ce scrieti codul;
  - Va ajuta prin afisarea tuturor metodelor unei clase atunci cand puneti punct dupa instanta acelei clase.
    - Nu mai e nevoie sa rasfoiti documentatia…





Java - RobotNou/src/Cata/Cata1.java - Eclipse Platform

File Edit Source Refactor Navigate Search Project Jacobe Run Window Help

Package Explorer

- AlgoritmiCulturali
- ClusteringPPSN
- ESVMs Fibro
- ParticuleStol
- PPSN
- RobotNou
  - src
    - Cata
      - Cata1.java
  - JRE System Library [jre1.5.0_16]
  - Referenced Libraries
- Socoteli
- SYNASC 2008
- X si 0

Outline

- Cata
- import declarations
  - robocode.*
  - java.awt.Color
  - java.awt.event.MouseEvent
- Cata1
  - run()
  - onScannedRobot(ScannedRobotEv
  - onHitByBullet(HitByBulletEvent)
  - onWin(WinEvent)
  - onBulletMissed(BulletMissedEvent)
  - onBulletHit(BulletHitEvent)
  - onMouseEntered(MouseEvent)
  - onMouseExited(MouseEvent)

parametersFile.txt | MyGame.java | *Cata1.java

```
package Cata;
import robocode.*;
import java.awt.Color;
import java.awt.event.MouseEvent;

/**
 * Robot1 - a robot by Cata
 */
public class Cata1 extends Robot
{
    /**
     * run: Robot1's default behavior
     */
    public void run() {
        setColors(Color.red,Color.blue,Color.green);
        System.out.println("Incepe batalia!!");
        while(true) {
            // Replace the next 4 lines with any behavior you would like
            ahead(100);
            turnGunRight(360);
            back(100);
            turnGunRight(360);
        }
    }

    /**
     * onScannedRobot: What to do when you see another robot
     */
    public void onScannedRobot(ScannedRobotEvent e) {
        fire(1);
    }

    /**
     * onHitByBullet: What to do when you're hit by a bullet
     */
    public void onHitByBullet(HitByBulletEvent e) {
        turnRight(e.getBearing());
    }

    public void onWin(WinEvent e)
    {
```

Problems | Declaration | Console

No consoles to display at this time.

Writable | Smart Insert | 12 : 38

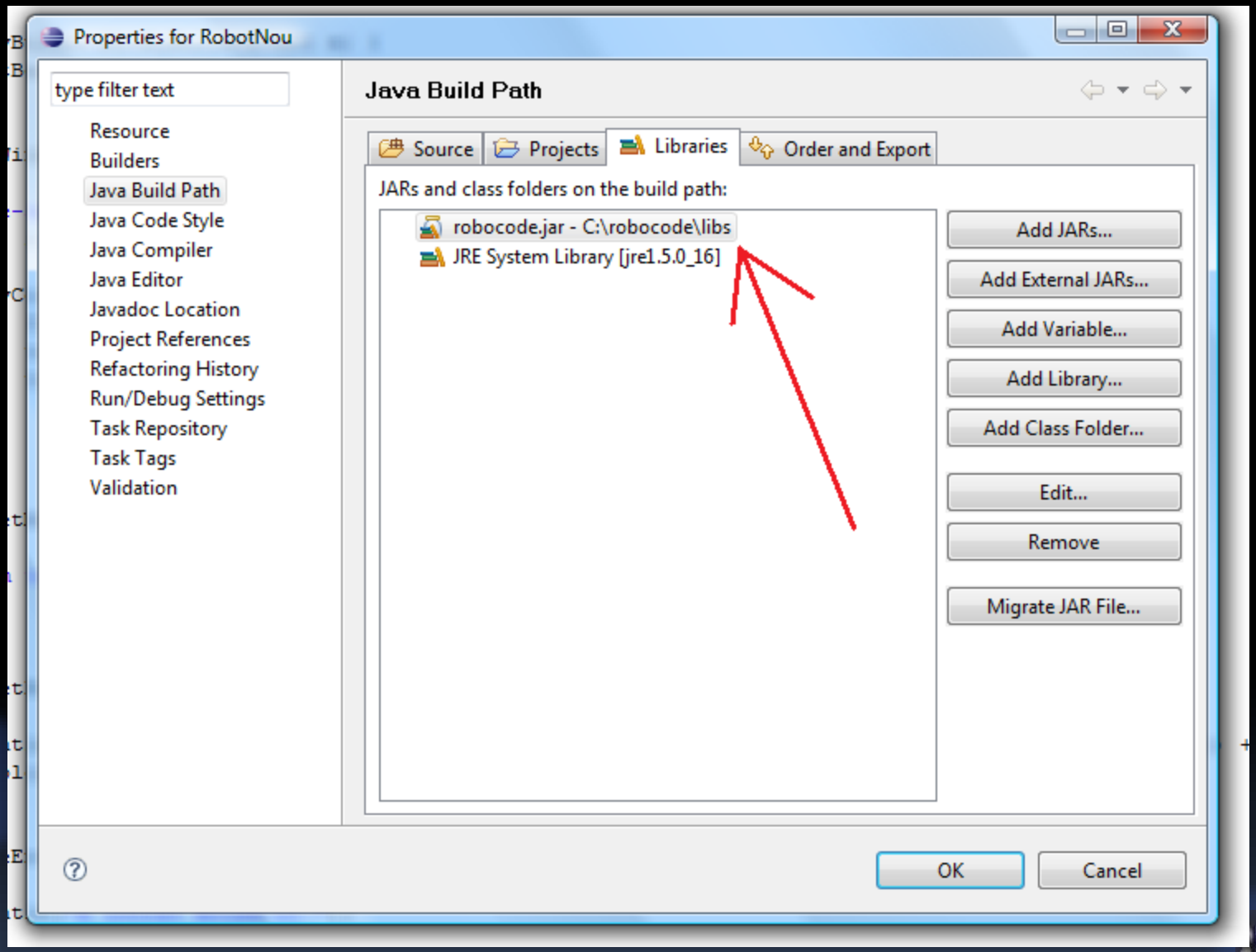
robocode_c.pdf -... | Robot (Robocode... | Microsoft Power... | Robocode | Robocode | Robot Editor | cs.Robot1 | Java - RobotNou/... | EN | 15:00

Cum scrieti codul pentru un robot in Eclipse

- Creati un nou proiect: File->New->Java Project
- Ii dati un nume, apoi salvati.
  - Atentie, cand adaugati la proiect un fisier java, acesta trebuie sa aiba acelasi nume cu clasa continuta in el (de exemplu, clasa Robot1 va fi scrisa in fisierul Robot1.java)
- Mergeti in meniul Eclipse la Project->Properties->Java Build Path->Libraries->Add external JARs si cautati Robocode.jar in locul unde ati instalat Robocode: daca ati pus chiar pe c:\, in c:\robocode\libs.





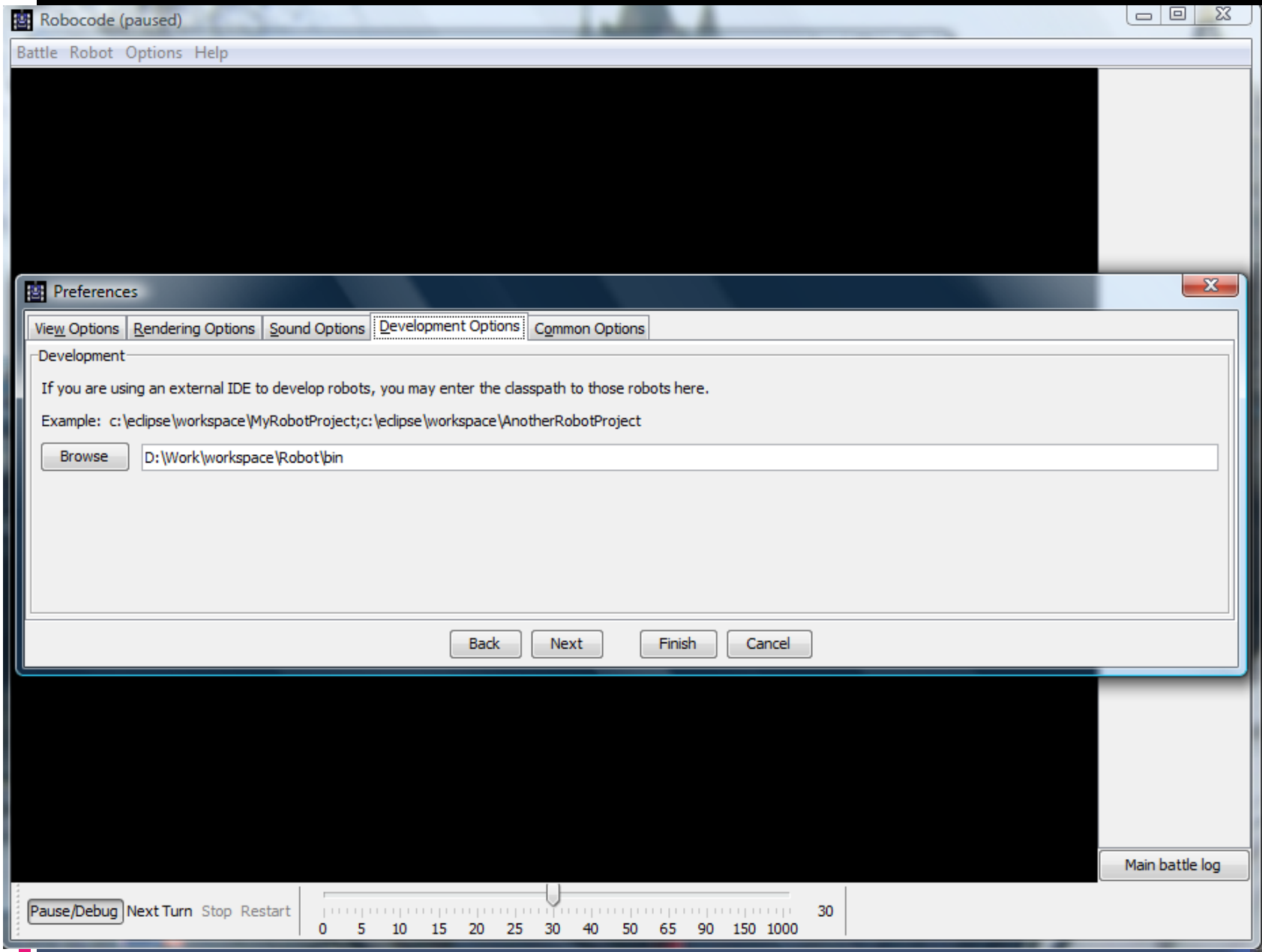

Properties for RobotNou
type filter text
Resource
Builders
Java Build Path
Java Code Style
Java Compiler
Java Editor
Javadoc Location
Project References
Refactoring History
Run/Debug Settings
Task Repository
Task Tags
Validation
Java Build Path
Source
Projects
Libraries
Order and Export
JARs and class folders on the build path:
robocode.jar - C:\robocode\libs
JRE System Library [jre1.5.0_16]
Add JARs...
Add External JARs...
Add Variable...
Add Library...
Add Class Folder...
Edit...
Remove
Migrate JAR File...
OK
Cancel

Cum scrieti codul pentru un robot in Eclipse

- Pentru a crea pachetul in Eclipse, mergeti in c:\robocode\robots si mutati pachetul creat acolo la inceput (cel care contine codul pentru robotul personal si este referit prin initialele date) in directorul workspace\Robot1\src in directorul eclipse.
- Compilati fisierul in Eclipse.
- Lansati Robocode de pe desktop si mergeti in meniul Options\Preferences\Development options pentru a ii da calea catre clasa generata acum de Eclipse.






Robocode (paused)
Battle Robot Options Help
Preferences
View Options
Rendering Options
Sound Options
Development Options
Common Options
Development
If you are using an external IDE to develop robots, you may enter the classpath to those robots here.
Example: c:\eclipse\workspace\MyRobotProject;c:\eclipse\workspace\AnotherRobotProject
Browse
D:\Work\workspace\Robot\bin
Back
Next
Finish
Cancel
Main battle log
Pause/Debug
Next Turn
Stop
Restart
0 5 10 15 20 25 30 40 50 65 90 150 1000
30

Mai multe despre Robocode

- http://robocode.sourceforge.net/
  - Stiri, downloads, documentatii, forumuri, competitii, clasamente etc.
- http://testwiki.roborumble.org/
  - Wikipedia despre Robocode.
- http://www.ibm.com/developerworks/java/library/j-robocode/
- sau doar dati “Robocode” intr-un motor de cautare.