

# Robocode

Dr. Günter Kolousek

2011-06-25

# Outline

**1** Überblick

**2** Spielphysik

**3** Programmierung

# Was

- **Programmierspiel**
- Roboter werden programmiert
- ... in Java
- ... und kämpfen gegeneinander
- eigene Entwicklungsumgebung

# Installation und Start

- Installation

```
java -jar robocode-1.7.3.0-setup.jar
```

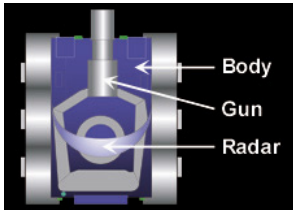
- Robocode ausführen

```
~/robocode/robocode.sh
```

# Aufbau eines Roboters 1

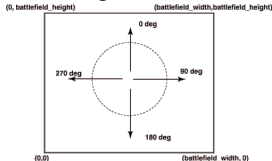
- Unterbau (Body)
  - trägt die Kanone mit dem Radar
  - vorwärts bzw. rückwärts fahren
  - nach links bzw. rechts drehen
- Gun (Kanone)
  - ist am Body montiert
  - nach links bzw. rechts drehen
- Radar
  - montiert auf der Kanone
  - nach links bzw. rechts drehen
  - generiert `onScannedRobot` Ereignisse (Event)

# Aufbau eines Roboters 2



# Koordinaten und Richtungen

- Kartesisches Koordinatensystem:  $(0, 0)$  links unten
- Einheit: Pixel, aber Gleitkommazahl!
- Roboter können Spielfeld nicht verlassen
- Richtungen
  - im Uhrzeigersinn
  - 0 deg (Grad) schaut nach Norden, 90 nach Osten, . . .
  - Richtungen im Überblick



# Zeit, Geschwindigkeit, Beschleunigung, Weg

time, velocity, acceleration, distance

- Zeit wird in *ticks* gemessen
- Jeder Roboter bekommt 1 tick je Runde (turn)
- 1 tick = 1 turn
- Beschleunigung: 1 pixel/turn/turn
- Verzögerung (Bremsen): 2 pixel/turn/turn
- Geschwindigkeit:  $v = at$  (Maximum: 8 pixel/turn)
- Weg:  $s = vt$



# Max. Rotation

- ... des Roboters:  $(10 - 0.75 * \text{abs}(\text{velocity}))$  deg / turn
- ... der Kanone: 20 deg / turn (zusätzlich zur Rotation des Roboters)
- ... des Radars: 45 deg / turn (zusätzlich zur Rotation der Kanone)

# Bullets

- Schaden:  $4 * \text{firepower}$ 
  - Wenn  $\text{firepower} > 1$ , dann zusätzlicher Schaden:  
 $2 * (\text{firepower} - 1)$
- Geschwindigkeit:  $20 - 3 * \text{firepower}$
- GunHeat:  $1 + \text{firepower}/5$ 
  - $\text{gunHeat} > 0$ : kein Feuern möglich
  - $\text{gunHeat}$  wird um 0.1 je Turn weniger
- gewonnene Power bei Treffer:  $3 * \text{firepower}$

# Kollisionen

- mit anderen Robotern: jeder 0.6
- mit der Wand (nur bei AdvancedRobots):  
 $\text{abs(velocity)} * 0.5 - 1$  (aber nicht kleiner als 0)

# Ablauf

- 1 Ansicht wird neu gezeichnet
- 2 Jeder Roboter führt seinen Programmcode aus (bis Aktion, dann Pause)
- 3 Zeit wird aktualisiert ( $\text{time} = \text{time} + 1$ )
- 4 Alle Bullets bewegen sich und es wird auf Kollisionen überprüft
- 5 Jeder Roboter bewegt sich (gun, radar, heading, acceleration, velocity, distance)
- 6 Jeder Roboter führt Scans aus
- 7 Jeder Roboter führt Aktion aus
- 8 Jeder Roboter fragt Ereignisse ab (event queue)

# Neuer Roboter

- 1 Robocode: Robot -> Editor
- 2 Robot Editor: File -> New Robot
- 3 Name beginnt mit Großbuchstaben, z.B.: SimpleRobo
- 4 Paketname beginnt üblicherweise mit Kleinbuchstabe, z.B.:  
ko

## Sourcecode 1

```
package ko;
import robocode.*;

public class SimpleRobo extends JuniorRobot {
    public void run() {
        // Initialization should be put here:
        // body, gun, radar, bullet, scan_arc
        setColors(orange,blue,white,yellow,black);

        while(true) {
            // Replace the next 4 lines!
            ahead(100);
            turnGunRight(360);
            back(100);
            turnGunRight(360);
        }
    }
}
```

## Sourcecode 2

```
public void onScannedRobot() {  
    // Power zwischen 0.1 (min) und 3 (max)  
    fire(1);  
}  
  
public void onHitByBullet() {  
    // Replace the next line!  
    back(10);  
}  
  
public void onHitWall() {  
    // Replace the next line!  
    back(20);  
}  
}
```

# JuniorRobot

- einfachster Roboter
- Instanzvariablen zum Zugriff auf Eigenschaften
  - z.B.: `scannedBearing...` Winkel zum nächsten Roboter relativ zum Body
    - `bearing` (Peilung)
- alle Methoden blockieren bis abgeschlossen
- ... und benötigen einen `turn`
- ... aber: setzen der Farben geht sofort und benötigt keine Zeit



# Übersetzen und ausführen

- Javaprogramm muss in Zwischencode übersetzt werden
- dann Battle erstellen
- Vorgang:
  - 1 Robot Editor: Compiler -> Compile
  - 2 Robocode: Battle -> New
  - 3 Roboter hinzufügen
  - 4 -> Start Battle
  - 5 Wenn fertig, dann Resultate ansehen. . .
- Neue (fremde) Roboter in das folgende Verzeichnis kopieren:  
`~/robocode/robots/<package>`

# Start

- Los geht's!
- Weitere Doku: `~/robocode/javadoc/index.html`