

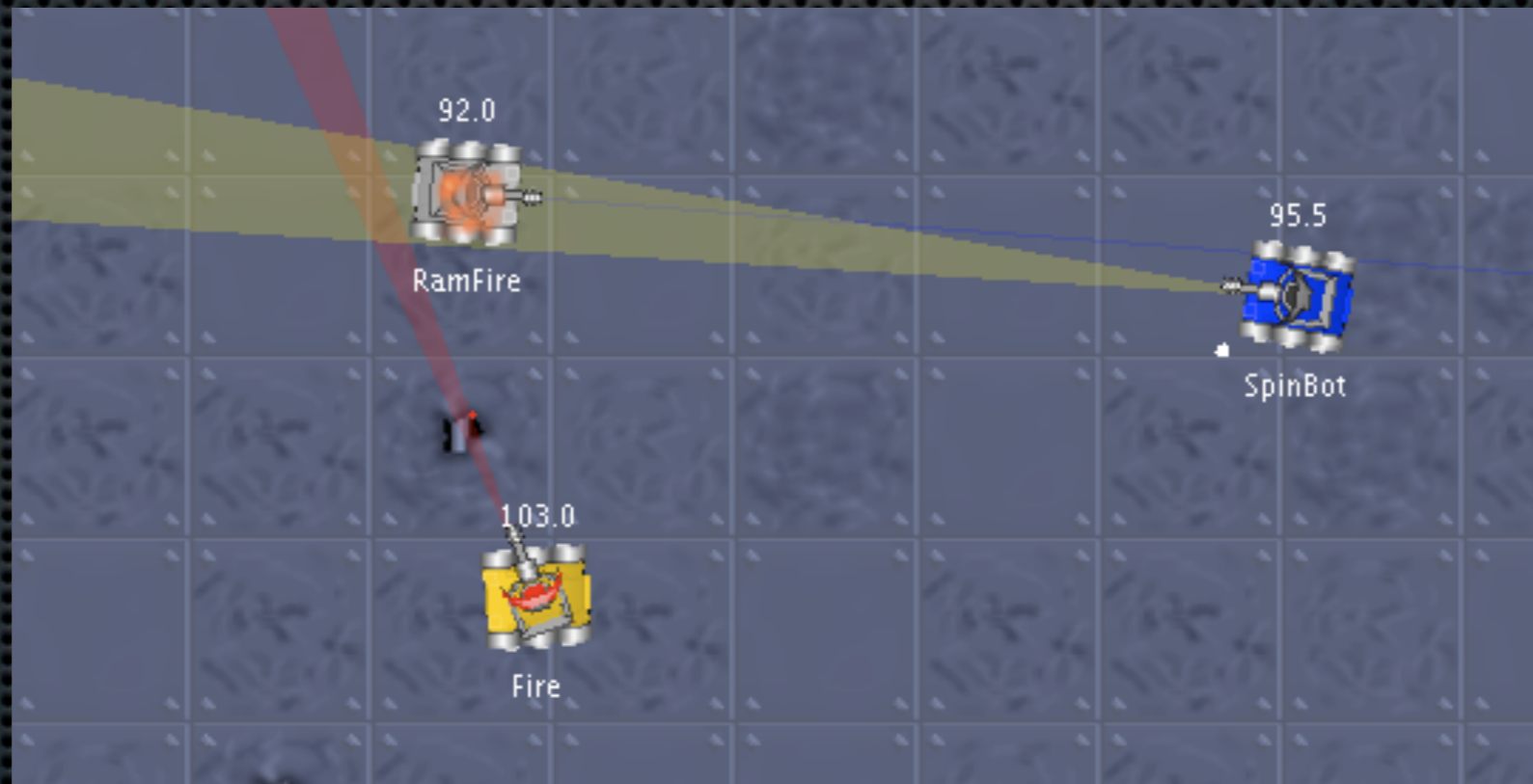
RoboCode Tutorial

Cholwich Nattee

School of ICT, SIIT, TU

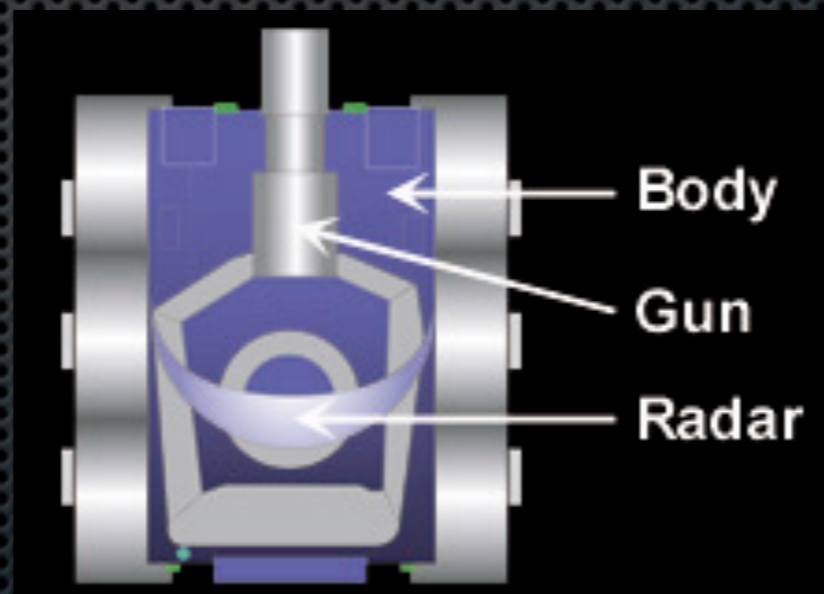
What is RoboCode?

- ✦ RoboCode is a **programming game**.
- ✦ **Goal:** to create a robot to compete in a battle field.



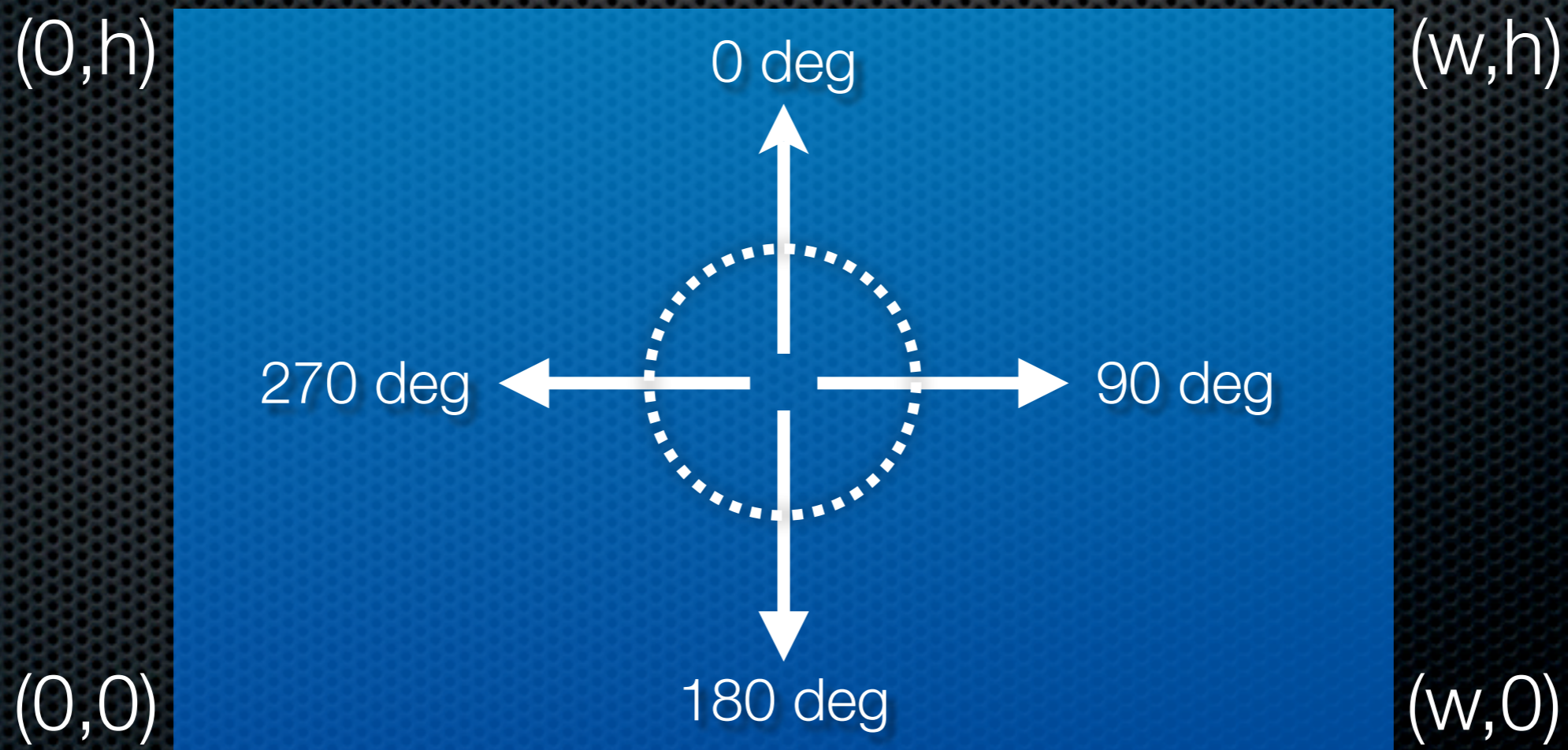
A Robot

- ✦ A **Robot** (36×36 pixels) = **Body** + **Gun** + **Radar**
 - ✦ **Body** for moving itself
 - ✦ **Gun** for firing energy bullets
 - ✦ **Radar** for detecting other robots



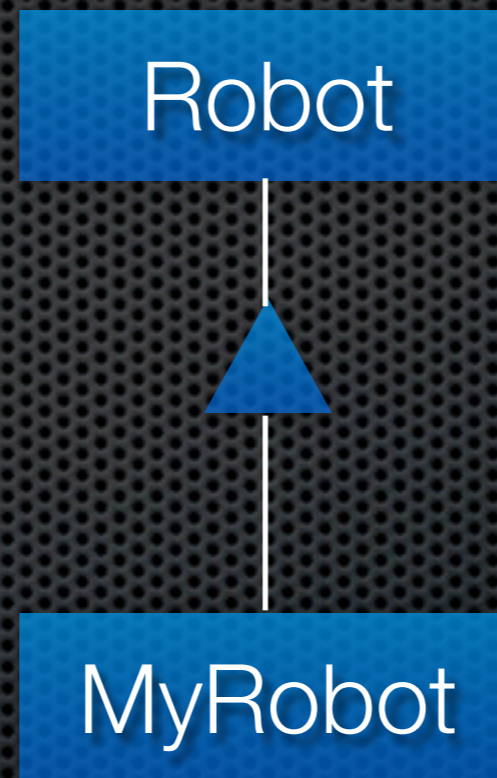
Game Physics: Battle Field

- **Cartesian coordinate system:** $(0,0)$ at the bottom left
- **Clockwise direction:** 0 degree towards “North”



Creating Your First Robot

- A Robot = A Java Class extending “Robot”



Creating Your First Robot

```
import robocode.*;
public class MyRobot extends Robot {
    public void run() {
        while(true) {
            ahead(100);
            turnGunRight(360);
        }
    }
    public void onScannedRobot(ScannedRobotEvent e) {
        fire(1);
    }
}
```

Game Physics: Time/Dist.

- ✦ **Time:** 1 turn = 1 tick, 1 tick depends on TPS
- ✦ **Acceleration:**
 - ✦ 1 pixel/turn for accelerating,
 - ✦ 2 pixel/turn for decelerating
- ✦ **Velocity:** $\text{velocity} = \text{acceleration} \times \text{time}$
- ✦ **Distance:** $\text{distance} = \text{velocity} \times \text{time}$; measured in pixels

Robot Movement

- ✦ **ahead(double distance)**
 - ✦ move forward for a specified distance
- ✦ **back(double distance)**
 - ✦ move backward for a specified distance

Rotation

- ✦ **Body Rotation**

- ✦ turnLeft(double deg) / turnRight(double deg)

- ✦ **Gun Rotation**

- ✦ turnGunLeft(double deg) / turnGunRight(double deg)

- ✦ **Radar Rotation**

- ✦ turnRadarLeft(double deg) / turnRadarRight(double deg)

Game Physics: Rotation

- ✦ **Body rotation:**

- ✦ $10 - 0.75 \times \text{abs}(\text{velocity}) \text{ deg/turn}$

- ✦ **Gun rotation:**

- ✦ $20 + \text{body_rotation} \text{ deg/turn}$

- ✦ **Radar rotation:**

- ✦ $45 + \text{gun_rotation} \text{ deg/turn}$

Bullet

- ✦ **fire(double power)**
 - ✦ fire a bullet with a specified power
- ✦ **fireBullet(double power)**
 - ✦ fire a bullet with a specified power
 - ✦ return an object of class “Bullet”

Game Physics: Bullets

- ✦ **Damage:** $4 \times \text{firepower} + [2 \times (\text{firepower} - 1) \text{ if } \text{firepower} > 1]$
- ✦ **Velocity:** $20 - 3 \times \text{firepower}$
- ✦ **GunHeat:** $1 + \text{firepower} / 5$
 - ✦ The gun does not work if $\text{GunHeat} > 0$
 - ✦ GunHeat is decreased by GunCoolingRate per turn
- ✦ **Power returned on hit:** $3 \times \text{firepower}$

Game Physics: Collisions

- ✦ **With Another Robot:**

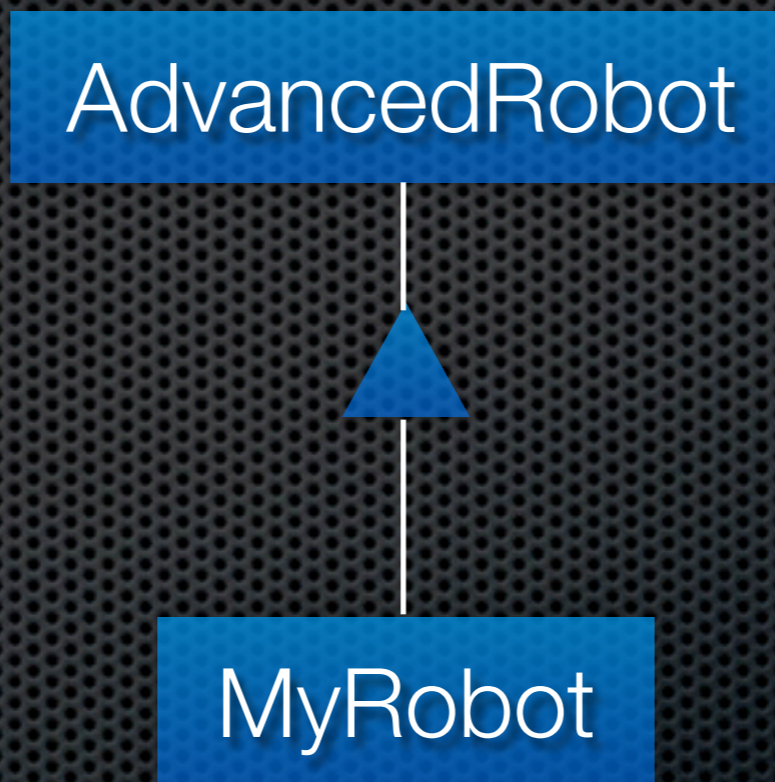
- ✦ Each robot takes 0.6 damage

- ✦ **With Wall**

- ✦ 0 for “Robot”
- ✦ $\max(\text{abs}(\text{velocity}) * 0.5 - 1, 0)$ for “AdvancedRobot”

AdvancedRobot

- More methods to control your robot



Detectable Events

- ✦ onBulletHit
- ✦ onBulletHitBullet
- ✦ onBulletMissed
- ✦ onDeath
- ✦ onHitByBullet
- ✦ onHitRobot
- ✦ onHitWall
- ✦ onRobotDeath
- ✦ onStatus
- ✦ onWin

Our Rule

Initial Energy = 100

Battle Field = 800×600

Gun Cooling Rate = 0.1

Our Tournament

- ✦ Qualifying Round - Round-robin with all teams
 - ✦ Win = 2, Draw=1, Loss=0
 - ✦ 3 games per match
- ✦ Final Round
 - ✦ Knock-out
 - ✦ 7 games per match