

# For Friday

- Finish Becker, chapter 2
- Recommended practice problems: chapter 2, problems 4-8

Questions before the quiz?

# Quiz

# Extended Class Pattern

# Constructors

- What's the purpose of a constructor?

# Writing Constructors

```
public class ExperimentRobot extends Robot
{
    // A constructor to initialize the ExperimentRobot
    public ExperimentRobot(City aCity, int aStreet, int
        anAvenue, Direction aDirection)
    { super(aCity, aStreet, anAvenue, aDirection);
    }
    // Another constructor to initialize the ExperimentRobot
    to be in a standard position.
    public ExperimentRobot(City aCity)
    { super(aCity, 0, 0, Direction.EAST);
    }
    // The new services offered by an ExperimentRobot will be
    inserted here.
}
```

Usage:

```
ExperimentRobot lisa = new ExperimentRobot(austin, 3, 2,
    Direction.SOUTH);
ExperimentRobot larry = new ExperimentRobot(austin);
```

# Creating New Services

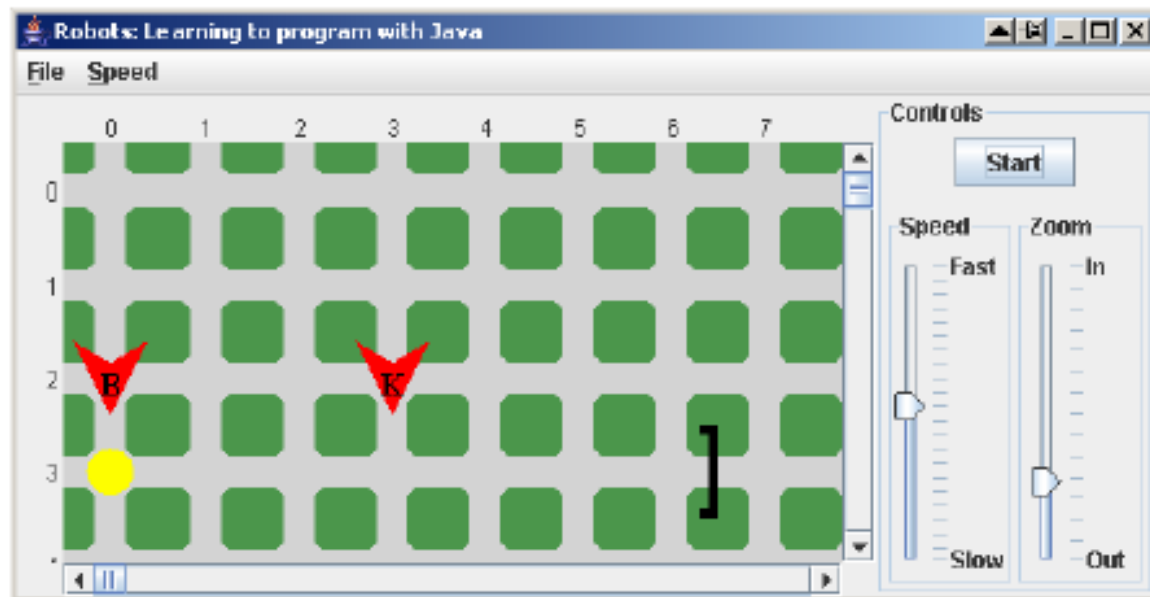
# Flow of Control

- What happens when we actually call a method?

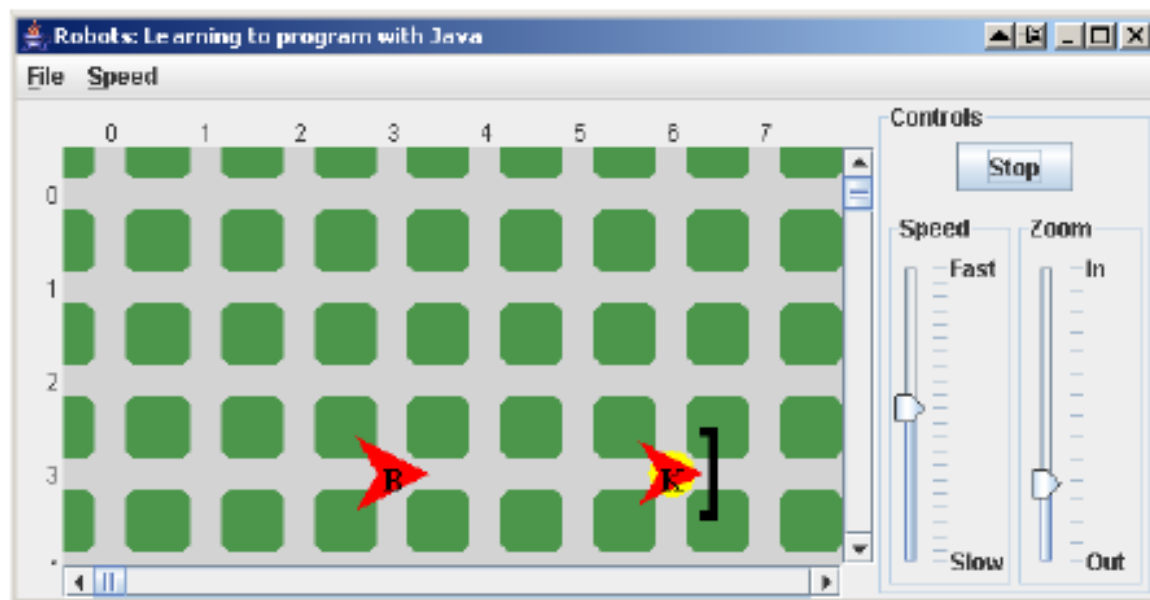
# RobotSE

- More capable version of Robot
- Extension vs. Modification

## Two robots running a “relay.”



Initial Situation

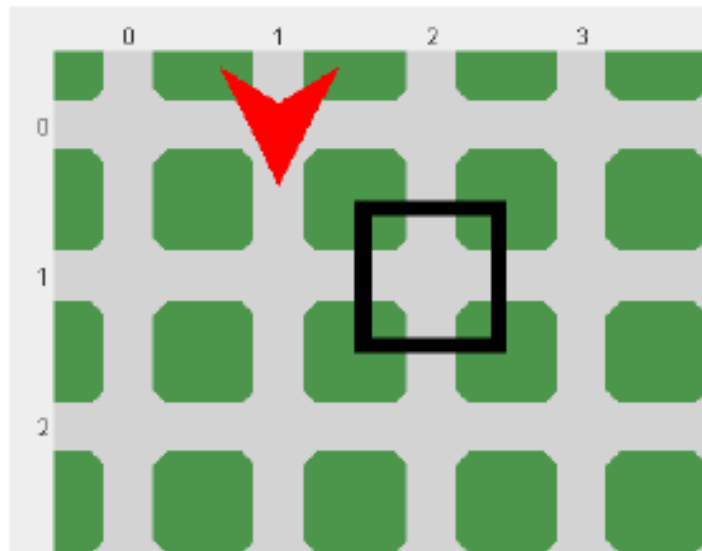


Final Situation

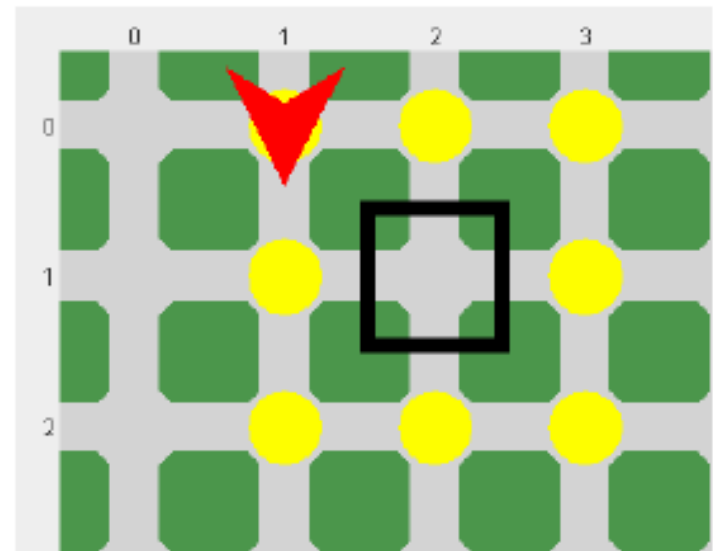
“B” picks up the baton and takes it to “K”, who finishes the race.

# Planting Flowers Again

- We're going to redo the planting flowers problem with two modifications:
  - Create and use a **GardenerBot** that has a method **plantFlowers**, which plants all the flowers.
  - Create an extended version of City called **Garden** that automatically includes the four walls.



Initial Situation



Final Situation

# The Importance of Style

```
import becker.robots.*; public class
ExperimentRobot extends Robot {public
ExperimentRobot(City aCity, int aStreet,
int anAvenue, Direction aDirection) {
super(aCity, aStreet, anAvenue,
aDirection);} public void turnAround() {
this.turnLeft(); this.turnLeft(); } public
void move3(){this.move(); this.move();
this.move(); } public void turnRight()
{this.turnAround(); this.turnLeft(); }}
```

# White Space and Indentation

- Start each statement on a new line.
- Include blank lines between blocks of code for different purposes (ideally with a comment to indicate what the following block of code does)
- Line up curly braces.
- Indent consistently
- Note: Eclipse will do most of this for you.

# Identifiers

- The rules
- The conventions

# Comments

- Single-line
- Multi-line
- Document

# Meaning and Correctness

- What if move3 were defined like this:

```
public void move3 ()  
{  
    this.turnLeft ();  
    this.pickThing ();  
    this.turnLeft ();  
}
```

# Modifying Inherited Methods

- Some people “show off” when they move. They might be loud or swagger or sway their hips or .... It seems like they can’t move in any other way. If they move, they show off.
- Create a **ShowOff** robot that turns magenta each time it moves. When it is standing still or turning (that is, not **move**-ing) it should continue to be red.

# ShowOff Robot

```
import becker.robots.*;
public class ShowOffMain
{
    public static void main(String[ ] args)
    {
        City ny = new City();
        ShowOff karel = new ShowOff(ny, 2, 3,
            Direction.SOUTH);
        karel.turnLeft(); // red
        karel.move(); // magenta
        karel.move(); // magenta
        karel.turnLeft(); // red
        karel.move(); // magenta
    }
}
```