Types of Programs

Review of the types of programs we have created.

Overview

- Program types
  - Using basic robots (no extra instructions)
    - Just one source file (containing a main method)
    - can have more than one robot
  - Creating new Robots
    - requires a source file for each new type of robot
    - And one containing the main method
- When must we specify the name of the robot?
- Using methods

A program may have just one source file

```
public class Test
{
   public static void main(String [] args)
   {
      World test = new World();
      Robot bobo = new Robot();
      test.add(bobo);
      bobo.move();
      bobo.turnLeft();
      bobo.move();
   }
}
```

Test.java

A world with two robots

```
public class Test2
{
   public static void main(String [] args)
   {
      World crowded = new World();
      Robot jack = new Robot();
      Robot jill = new Robot();
      crowded.add(jack, 7, 7, "north");
      crowded.add(jill);
      jack.move();
      jill.turnLeft();
      jill.move();
   }
}
```

Create a New Type of Robot

```
public class RightTurner extends Robot
{
   void turnRight()
   {
      turnLeft();
      turnLeft();
      turnLeft();
   }
}
```

Another class is needed to create the world and use this robot's methods.

Use a New Type of Robot

```
public class TestRightTurner
{
   public static void main(String [] args)
   {
      World us = new World();
      RightTurner bush = new RightTurner();
      us.add(bush, 7, 7, "north");

      bush.turnLeft();
      bush.move();
      bush.turnRight(); // use the new instruction
      bush.move();
   }
}
```

Different Types of Robot

```
public class TestRightTurner
{
   public static void main(String [] args)
   {
      World us = new World();
      RightTurner bush = new RightTurner();
      Robot osama = new Robot();
      us.add(osama, 7, 7, "north");
      us.add(bush, 7, 8, "south");
      osama.turnLeft();
      osama.move();
      bush.turnRight();
   }
}
```

But ...

```
public class TestRightTurner
{
   public static void main(String [] args)
   {
      World us = new World();
      RightTurner bush = new RightTurner();
      us.add(bush, 7, 7, "north");

      bush.turnLeft();
      bush.move();
      bush.turnRight(); // use the new instruction
      bush.move();
   }
}
```

But

```
public class RightTurner extends Robot
{
   void turnRight()
   {
      turnLeft();
      turnLeft();
      turnLeft();
   }
}
```

RightTurner is a type of Robot and so the turnLeft refers to the Robot itself.

Using methods

```
public class EightWalker extends Robot
{
    void move8()
    {
        move4();
        move4();
    }
    void move4()
    {
        move();
        move();
        move();
        move();
    }
}
```

move8 is defined in terms of the move4 method.

Using methods2

```
public class EightWalker extends Robot
{
   void move8()
   {
      move4();
      move4();
   }
   void move4()
   {
      move2();
      move2();
   }
   void move2()
   {
      move();
      move();
   }
}
```

move8 is defined in terms of the move4 method which is itself defined in terms of move2

Summary

- Program types
  - Using basic robots (no extra instructions)
    - Just one source file (containing a main method)
    - can have more than one robot
  - Creating new Robots
    - requires a source file for each new type of robot
    - And one containing the main method
- When must we specify the name of the robot?
- Using methods