“Open Robotics”
A system for Objectifying and Simplifying Your Robot Code

Presented by:

Harish Thiagarajan, Lynbrook High School

Outline

- What is OOP?
- “Karel J. Robot” API
- An overview of leJOS
- Drag and Drop (DnD) coding system
- Open Robotics: Combining a DnD solution with the simplicity of Objectified code to help beginners

An Intro to OOP

- Object Oriented Programming
- Create class definitions that act as “templates”
- Class definitions consist of:
  - An object’s attributes/traits
  - Actions the object can perform (methods)
  - Methods that can access attributes/traits

Karel

- A basic Java API that simulates a robot that moves around in a grid-like environment

Simple Karel primitives

- move()
- turnLeft() – I will explain later why there is no turnRight()
- putBeeper() – Beepers are small black circles that Karel may interact with
- pickBeeper()
- frontIsClear()
- facingNorth()

FIRST®

“Complex” Karel methods

- What is “turnRight()” if turnLeft() turns Karel 90 degrees?
  - Simply put, turnRight() is actually 3 turnLeft()’s executed consecutively
  - (Will this kill you during the competition? What can we do?)
- Adding complexity – primitive methods and conditionals can be combined to allow complex behavior such as maze traversal or even home vacuuming

Inheritance and Maintenance

- Easily reuse previously written code
- Ability to “inherit” the methods of a “superclass” in order to create a “subclass”
- e.g. Karel1 implemented the turnRight() method, then Karel2, the subclass, implemented the turnRightTwice() method using the definition of turnRight() from Karel1 rather than defining from scratch
- Also, if turnRight() from Karel1 is improved, then turnRightTwice() in Karel2 automatically benefits

leJOS

- Created by programmers who wanted to be able to program Mindstorm robots using Java
- Useful because it allows Java (an OOP language) to be implemented in programming robots
- Open Robotics code will be generated based on this API

Drag and Drop

- The ability for the code to be generated and modified with minimal worry about syntax
- Involves “dragging” code blocks to anywhere in the code body and entering the core details of the syntax, allowing the coding environment to take care of the rest
- Hides complexity and allows user to focus on their tasks
- For example, handles “turn right” and automatically generates efficient code

Best of both worlds

- Combine advantages of OOP and DnD
- The simplicity of commands such as move() and turnLeft() as opposed to SetMotor()
- Simple commands can be aggregated to form more complex commands
- The convenience of not needing to think about code syntax via a Drag and Drop environment

Advantages of existing DnD

- Intelitek's easyC introduced in 2005 for VEX and subsequently in 2006 also for FRC
- EasyC is a good example of a Drag and Drop implementation
- Allows simple and interactive editing of the code body

Disadvantages of existing DnD

- Although most of the syntax is handled, it may still be hard for novice programmers to understand what they need to code in order for the program to run as intended

Disadvantages of OOP

- Need to know OOP principles
- Must have programming expertise in Java or other OOP language to fully benefit

FIRST®

The “Open Robotics” solution

- A fusion between the simplicity of OOP and the convenience of DnD
- No syntax worries
- Think in terms of “commands” -- not OOP
- Do not have to worry about processor primitives like setMotor or the more complex pwm1=pwm8=127 (or was that 64?)
- Can “practice” with simple processor at home
- Then retarget for a more complex processor with same underlying “logic”

FIRST®

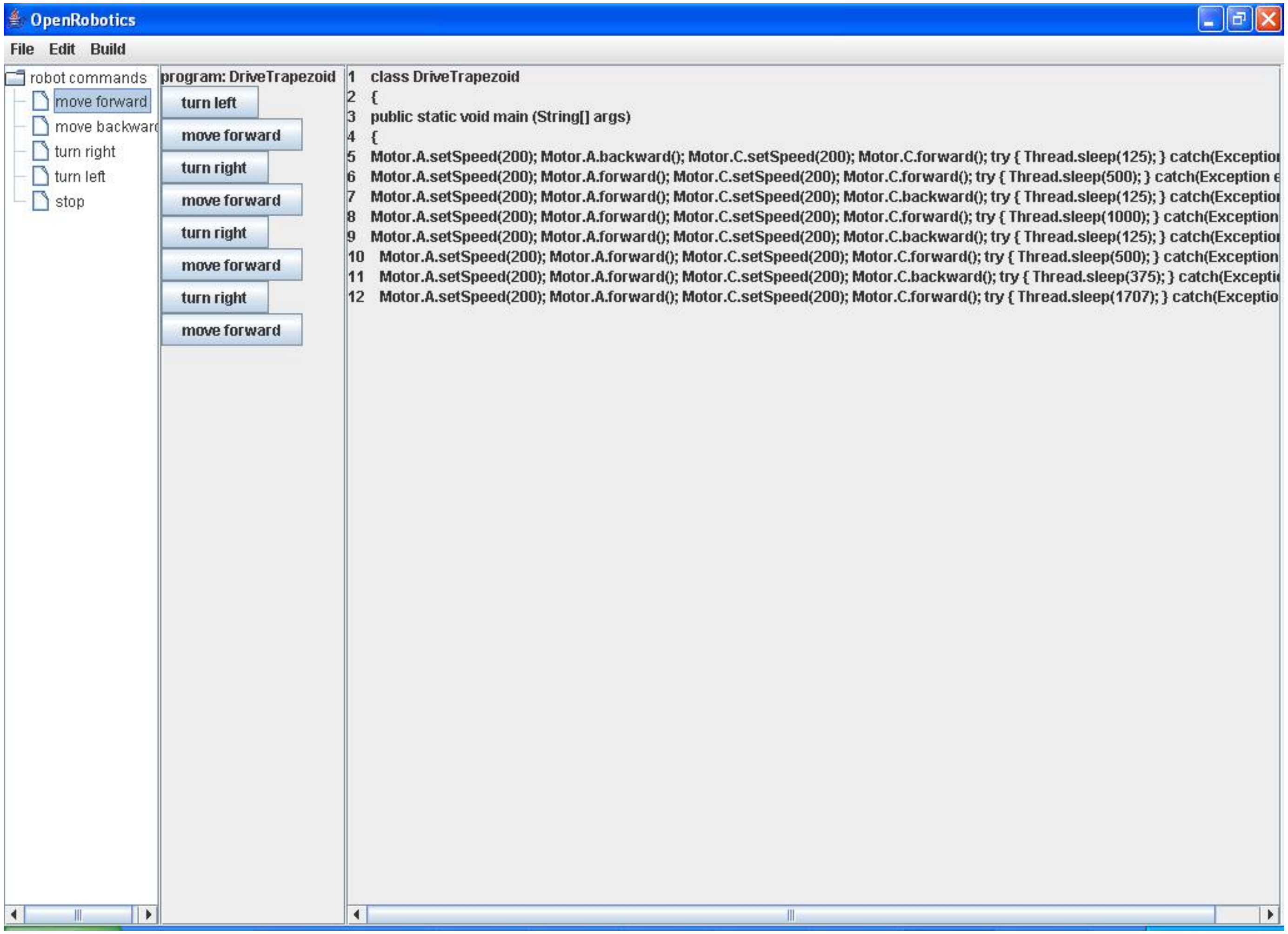
Conclusion

- OOP or “Object Oriented Programming” involves writing code as objects with properties and actions
- The Karel API simulates this concept with basic methods such as move() and turnLeft()
- Drag and Drop facilitates the development of code by handling the syntax and structure
- The “Open Robotics” solution combines the benefits of both, especially for beginners

OpenRobotics Prototype Screenshots

- In this example, the code is for Lejos
- Next, note how “angles” are translated to more complex setSpeed() and Thread.sleep()

OpenRobotics Prototype Screenshots



FIRST®

Scope for Further Work

- Currently hosting a software project on Source Forge: “Open Robotics Drag and Drop Programming”
- Many aspects of API scope and DnD interactivity still need to be implemented
- Looking for developers to join Open Robotics – just sign up on SourceForge
- http://sourceforge.net/projects/openrobotics