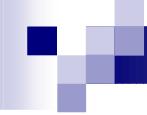


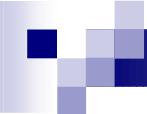


January 5th, 2005
Yuran Lu



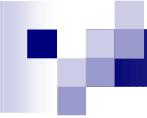
Agenda

- Getting Started
 - On the Server
 - Using the Documentation
 - Design Sequence
 - Tools
 - The Maslab API
- Design Principles
- Threading in Java



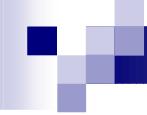
On the Server

- Put these lines in your .environment:
 - *add 6.186*
 - *add -f java_v1.5.0*
 - *setenv JAVA_HOME /mit/java_v1.5.0*
 - *setenv CLASSPATH /mit/6.186/2005/maslab.jar:*
- If you're Serverphobic, just ask for help.
You'll learn fast, and you'll be glad you did.



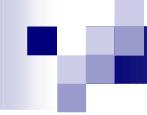
Using the Documentation

- Maslab API
- Java 1.5.0 API
- Sun's Java Tutorial
- Ed Faulkner's Java Reference
- All linked from SoftwareInfo page on wiki



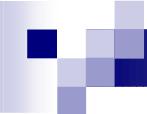
Design Sequence

- Open a text editor to edit a source code file:
 - *emacs MyExample.java*
- Write class declaration, and declarations for each of your methods, and annotate with comments
- Fill in source code
- Compile:
 - *javac MyExample.java*
 - This produces *MyExample.class* if successful
- Fix compile errors and repeat compilation until successful
- Run:
 - *java MyExample*
 - This searches the CLASSPATH for *MyExample.class* and executes it.



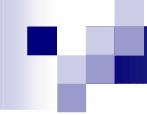
Tools

- CVS, subversion
- make, ant
- OrcSpy, BotClient
- Instructions all on SoftwareInfo page of wiki



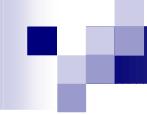
The Maslab API

- maslab.orc
- maslab.camera
- maslab.telemetry.channel



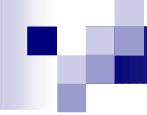
Agenda

- Getting Started
- Design Principles
 - Motivation
 - Modularity and the Design Process
 - Writing Good Specifications
 - Testing
 - Good Design Practices
- Threading in Java



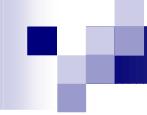
Design Principles - Motivation

- Coding a Maslab Robot is a formidable, multi-person project
- Making debugging easier
- Making sure different team member's code all work together
- Making sure one team member's changes doesn't break another team member's code



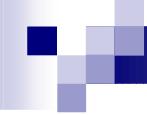
Modularity and the Design Process

- Modular Design
 - Provides abstraction
 - Gives up fine-control abilities, but makes code much more manageable
- The Design Process
 - Top-down vs. Bottom-up
 - Write out specifications for each module
 - Write code for modules
 - Test each module separately as it is being written
 - Test overall system for functionality



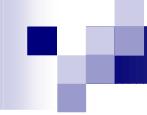
Modularity - An Example

- Start with most basic behaviors:
 - *DriveTowardBall*
 - *WallFollow*
 - *DriveToWallAndStop*
- Build up more complicated behaviors:
 - *HuntRedBalls*
 - *GoToLastRememberedBall*
 - *AlignAndDepositBall*
 - *WanderToGetUnstuck*
- Build highest-level behaviors:
 - *WinMaslab*



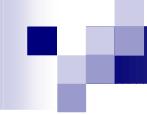
Writing Good Specifications

- This that should go into the specification:
 - Synopsis of classes and methods
 - How methods are called
 - Restrictions on argument values
 - The return value and effect of calling the method
- What shouldn't go into the specification:
 - How code is implemented
 - Long paragraphs of text



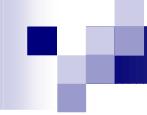
Testing

- Test each module separately
- Test overall system
- Test special cases
- Come up with test cases before coding, or have a different team member do testing
- Using the main() method
- Unit testing



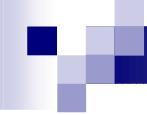
Good Design Practices

- EXTREMELY IMPORTANT!
- Thou shalt Test Constantly
- Start small, build up
- Modularity
- Avoid over-abstraction
- Back up code
 - Keep multiple versions backed up
 - Keep separate backups off of the robot computer



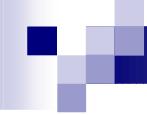
Agenda

- Getting Started
- Design Principles
- Threading in Java
 - Motivation
 - Using Threading
 - Synchronization



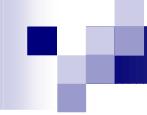
Motivation for Threading

- Ability to perform tasks in parallel
- If used properly, can make your robot run faster
- Different threads for:
 - Image capture and processing
 - Keeping a current map
 - Controlling the current motion behavior (Wandering, Ball-seeking, Obstacle Avoidance, etc.)
 - Higher-level strategic control



Using Threading

- Look in Sun's Java tutorial, or Ed Faulkner's Java reference
- Look at the Java API:
 - Thread, Runnable, wait(), notify(), sleep(), yield()
- Must take care to avoid deadlock



Synchronization in Threading

- Allows blocks of code to be mutually exclusive
- Writing to the same object from two threads at the same time will cause your program to break