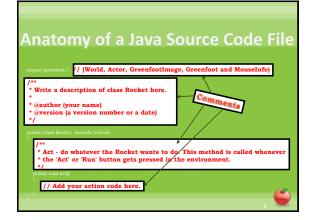
CSE 113 A

September 14 – 18, 2009

Announcements

- Lab 1 posted on course website due 10/2
- If you are having trouble logging into the computers in the lab or the Web-CAT website that we will be using for submission, please email me.
- Turn in signed last page of syllabus by 9/14.





Comments

- Comments are inserted by the programmer and ignored by the compiler.
- They explain to other humans reading the code the purpose of some of its parts.
- Comments can also be used to generate a special kind of documentation for Java source code files that we will discuss later on.

Anatomy of a Java Source Code File

a Definition

import greenfoot.*; // (World, Actor, GreenfootImage, Greenfoot and MouseInfo

/** * Write a description of class Rocket here. *

* @author (your name) * @version (a version number or a date)

public class Rocket extends Vehicle { /**

/ * Act - do whatever the Rocket wants to do. This method is called * whenever the 'Act' or 'Run' button gets pressed in the * environment. */

public void act()

// Add your action code here.

Class Definition (Template)

public class *Name*

ł

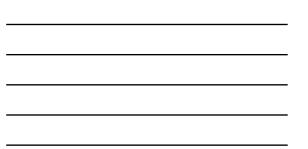
bublic and class are keywords in Java. They must be the first two things in your class definition.

Name is the name of the class – the programmer selects an appropriate name.

The {} show us the body of the class.

Anatomy of a Java Source Code File





Method Definition (Template)

public returnType methodName ()

- }____
- public is a repeat of the keyword we saw previously it must be the first word in your method definition.
- **returnType** is the type of information returned from the method void if nothing is returned.
- methodName is the name of the method the programmer selects an appropriate name
- () is the parameter list if no parameters are needed, then the () remain empty. If parameters are needed, they are listed by their type and also given a name.

The {} show us the body of the method – whe

Anatomy of a Java Source Code File

import greenfoot.*; // (World, Actor, GreenfootImage, Greenfoot and MouseInfo)

- * Write a description of class Rocket here
- * @author (your name) * @version (a version number or a da
- public class Rocket extends Vehicle

** Act - do whatever the Rocket wants to do. This method is called wheneve * the 'Act' or 'Run' button gets pressed in the environment.

turn(14);

Method Call (Template)

nameOfMethod (value);

- nameOfMethod is the name of the method you are calling (invoking)
- **palue** is the actual value for a parameter. If no parameters are indicated for the method, then value is omitted and the parentheses remain empty.

Decisions

- One of the things programs often do is make decisions on what to do next.
- We saw this when we wanted to do something speci if our actor reached the end of the world.
- There are many ways to get Java programs to react differently under different circumstances – we are only going to focus on if-statements this semester as a mechanism of choice (selection) in our programs.

If-statement (Template)

<mark>if</mark> (condition) {

- if is the Java keyword indicating that we are using a selection (choice) statement
- **condition** is inside the () and is always a boolean expression
- If the condition in the parentheses is true, we execute the code inside the {}. Otherwise, we skip over the code in the {} and move on to the next lines of code.

Randomizing Behavior

- Output: Up until this point, the actor behaved the exact sam way each time the act method was called.
 - It checked to see if it was at the edge of the world. If it was, it turned.
 - It checked to see if it could see another car. If it did, it turned.
- O It moved.
- If we want to randomize the behavior of the actor, we need to insert code to do that.

