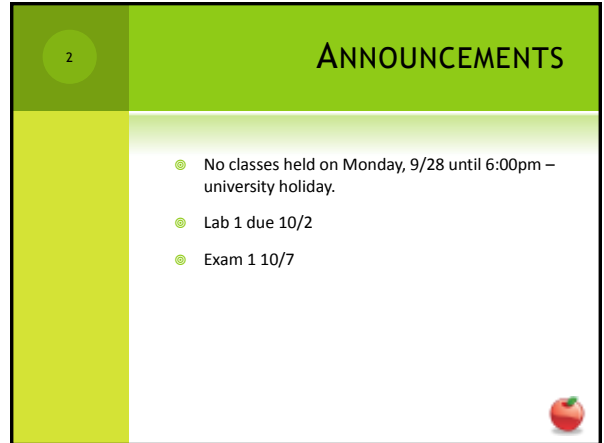


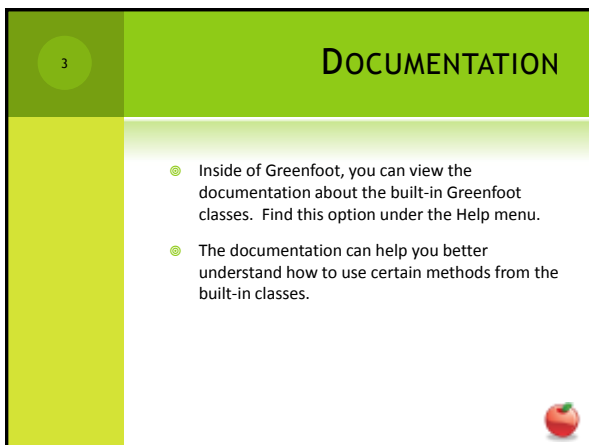

CSE 113 B
September 21 – 25, 2009



2

ANNOUNCEMENTS

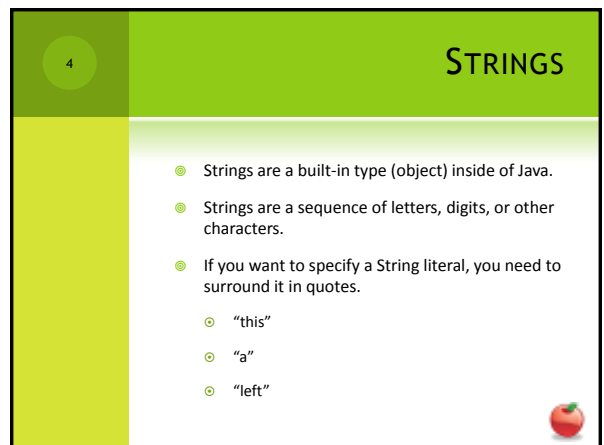

- No classes held on Monday, 9/28 until 6:00pm – university holiday.
- Lab 1 due 10/2
- Exam 1 10/7



3

DOCUMENTATION


- Inside of Greenfoot, you can view the documentation about the built-in Greenfoot classes. Find this option under the Help menu.
- The documentation can help you better understand how to use certain methods from the built-in classes.



4

STRINGS


- Strings are a built-in type (object) inside of Java.
- Strings are a sequence of letters, digits, or other characters.
- If you want to specify a String literal, you need to surround it in quotes.
 - "this"
 - "a"
 - "left"



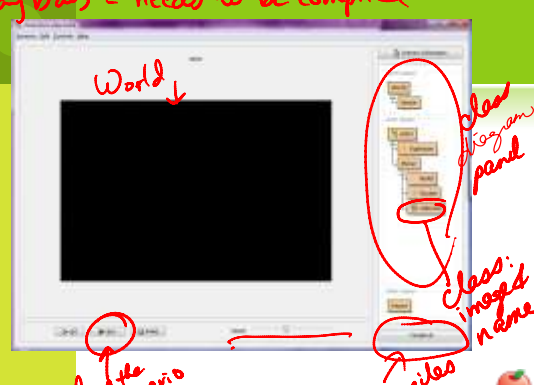
5

REVIEW

- The next several slides indicate review materials that were covered in class on Monday 9/21 and Wednesday 9/23. They incorporate the main ideas from Chapter 1 – 3 of the text.



gray bars = needs to be compiled




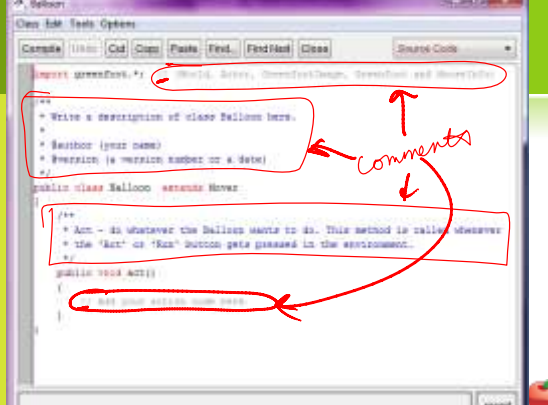
World ↓

class diagram panel

class: insert name

compiles

runs the scenario

comments

method definition


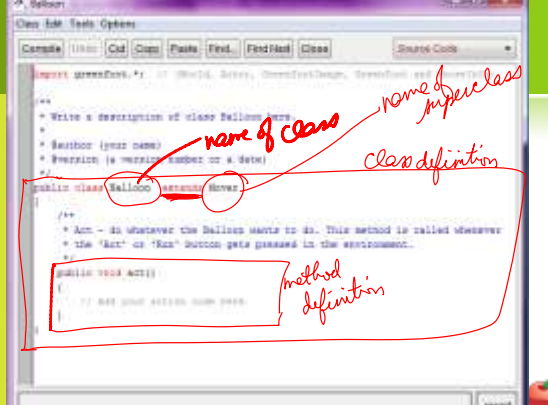
```

import javax.swing.*;

/**
 * Write a description of class Balloon here.
 * @author (your name)
 * @version (a version number or a date)
 */
public class Balloon extends JWindow {

    /**
     * Act - do whatever the Balloon wants to do. This method is called whenever
     * the "Act" or "Run" button gets pressed in the environment.
     */
    public void act() {
        // do your action here here
    }
}

```

name of superclass

name of class

class definition

method definition


```

import javax.swing.*;

/**
 * Write a description of class Balloon here.
 * @author (your name)
 * @version (a version number or a date)
 */
public class Balloon extends JWindow {

    /**
     * Act - do whatever the Balloon wants to do. This method is called whenever
     * the "Act" or "Run" button gets pressed in the environment.
     */
    public void act() {
        // do your action here here
    }
}

```



```

import greenfoot.*;

/**
 * Write a description of class Balloo here.
 *
 * @author (your name)
 * @version (a version number or a date)
 */
public class Balloo extends Mover {

    /**
     * Act - do whatever the Balloo wants to do. This method is called whenever
     * the 'Act' or 'Act+' button gets pressed in the environment.
     */
    public void act() {
        // your code here
    }
}

```

```

public class Greenfoot {

    /**
     * Act - do whatever the Car wants to do. This method is called whenever
     * the 'Act' or 'Act+' button gets pressed in the environment.
     */
    public void act() {
        // your code here
    }

    public void turn(int degrees) {
        // your code here
    }

    public void move(int dx, int dy) {
        // your code here
    }
}

```

11

Write the code for an act method that does the following:

- if hit edge of world turn between -30 and 30 degrees randomly
- if hits a Car, play sound "crash.wav" and stop the simulation
- 25% of time it should move
- 50% of the time it should turn *Sideways*

12

```

if (atWorldEdge())
{
    turn(Greenfoot, getNumber(60, 30));
}
if (canSee(Car.class))
{
    Greenfoot.playSound("crash.wav");
    Greenfoot.stop();
}
}


```

13

```

if (Greenfoot.getRandomNumber(100) < 25)
{
    move();
}
if (Greenfoot.getRandomNumber(100) < 50)
{
    turn(5);
}


```



14

QUESTIONS

- Use the previous slides as a study guide. The answer for the last question posed on the slides will be available the week of September 28th.



15


CONSTRUCTORS

- Constructors are special methods that are called each time an instance of a class is created.
- Constructors inside source code:


```

public SameNameAsClass()
{
}

```
- Note that there is no return type and the constructor will always have the same name as the name of the class.



16


CONSTRUCTORS

- Constructors are special methods that are called each time an instance of a class is created.
- Constructors inside source code:


```

public SameNameAsClass()
{
}

```
- Note that there is no return type and the constructor will always have the same name as the name of the class.



17

CONSTRUCTORS

- Inside the body of the constructor (inside the { }), you can do any of the same things you can do inside of other methods.
- Therefore, we can call methods from within a constructor.
- In our example, we call `super(560,560,1);`
- This is a call to a method named **super**. **super** is a keyword that actually indicates a call to the superclass' constructor.



18

ADDING OBJECTS TO THE WORLD

- Note that the **addObject** method of the world takes as its first parameter an Actor to be added.
- We need to create an actual instance to pass into this method.
- To create an object inside Java source code:
`new ConstructorName();`
- **new** is a keyword indicating that we are creating a new instance.
- **new** is followed by a call to the class' constructor. Values are inserted in the () if needed.



19

ADDING OBJECTS TO THE WORLD

- **addObject** also takes an x and y coordinate as parameters.
- We need to remember that in the coordinate system for graphics on computers, origin (0,0) is the upper left hand corner.
- The values of x increase as we move right on the screen and the values of y increase as we move down on the screen.

