Objective

To modify the “Get the Prize” lab (Lab 12) to include keyboard input.

4 //imports

. . .

9 **public class** PrizePanel **extends** JPanel  
 10 {

. . . //fields

16 **public** PrizePanel() //constructor

17 {

. . .

30 addKeyListener(**new** Key());

31 setFocusable(true);

. . .

44 }

45 //private listener classes

. . .

60 **private class** Key **extends** KeyAdapter  
 61 {

62 **public void** keyPressed(KeyEvent e)  
 63 {  
 64 **if**(e.getKeyCode() == KeyEvent.VK\_W)  
 65 ball.setY( ball.getY()-1 );   
 66 **if**(e.getKeyCode() == KeyEvent.VK\_Z)  
 67 ball.setY( ball.getY()+1 );   
 68 **if**(e.getKeyCode() == KeyEvent.VK\_A)  
 69 ball.setX( ball.getX()-1 );   
 70 **if**(e.getKeyCode() == KeyEvent.VK\_S)  
 71 ball.setX( ball.getX()+1 );   
 72 }  
 73 }

74 //public instance methods  
 . . .  
150 }

***Background***

Still another Listener is a KeyListener, which can be registered with any JPanel. Look at on Line 30 for an example.

Now each application can have three listeners, one each for the mouse, the keyboard, and the timer.

The Key object on Line 30 could either implement the KeyListener interface or extend the KeyAdapter class. If you implement KeyListener you’ll need to define three methods, even if you only use one of them. On the other hand, the KeyAdapter class provides default (empty) definitions for these methods, so only the methods that you actually use need to be defined.

Notice that Lines 64, 66, 68, and 70 call accessor methods of a mysterious KeyEvent object named e. e.getKeyCode() returns the key that was pressed. VK\_W, etc., are constants in the KeyEvent class. The result is that Lines 64 to 71 move the ball either 1 pixel up, down, left, or right.

Specification

Copy all the resource files from Lab16. Use the code above as a model so that the **arrow keys** move the **polkadot** **ten pixels** at a time. Do not allow the keys to move the polkadot outside the boundaries of the panel.

Load Filename Unit2\Lab17\Driver17.java. Notice that the driver calls the panel's method requestFocus. Alternatively, you can change the focus by clicking on the panel, which makes the keys work on that panel.

*Sample Run*

Unit2Lab17.jar

*Extensions*

1) Make a new folder for Pong. Copy into it all needed files from Lab14. Modify BumperPanel so that the keys move the bumper vertically.

2) Make a two-person Pong game. Keep score.

3) In Lab15, modify KarelPanel to use both mouse and keyboard input.

4) A website that students have found helpful is <http://www.faqs.org/docs/javap/index.html>