Lecture 27

Animation

Chapter 18
Animation

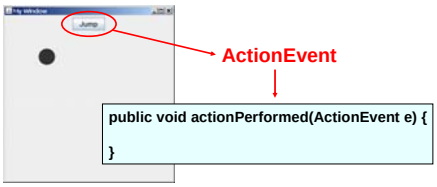
Introduction

We can make objects drawn on the screen appear to move by updating their positions slightly, and refreshing the screen, at regular short intervals.

Just like an old-fashioned flick book:

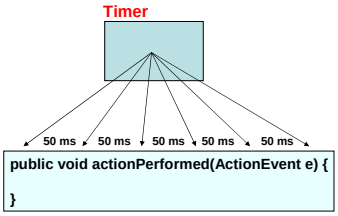
Animation with a JButton

- When we click a JButton, an ActionEvent is generated and the actionPerformed() method is called to handle that event.
- If the actionPerformed() method updates the position of an object drawn on the screen, and if we click the button quickly enough, then the object will appear to move



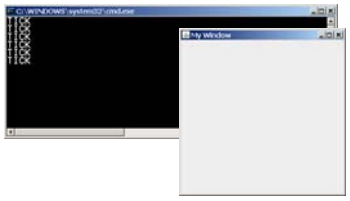
Using a Timer object

- The Timer class is a standard Java class defined in javax.swing.
- A Timer object can generate ActionEvents (i.e. call the actionPerformed() method) at regular intervals



Using a Timer object

- How do we create a Timer object?
- What methods can we call on a Timer object?
- To begin with, let's look at a simple example which uses a Timer object to print one line of output every second



```
public class MyJPanel extends JPanel {


      public MyJPanel() {


      }



      public void paintComponent(Graphics g) {
            super.paintComponent(g);

      }
}
```

```
public class MyJPanel extends JPanel implements ActionListener {


	public MyJPanel() {



	}

	public void actionPerformed(ActionEvent e) {

	}

	public void paintComponent(Graphics g) {
		super.paintComponent(g);

	}
}
```

```
public class MyJPanel extends JPanel implements ActionListener {

	private Timer t;

	public MyJPanel() {



	}

	public void actionPerformed(ActionEvent e) {

	}

	public void paintComponent(Graphics g) {
		super.paintComponent(g);

	}
}
```

```
public class MyJPanel extends JPanel implements ActionListener {

	private Timer t;

	public MyJPanel() {

		t = new Timer(1000, this);

	}

	public void actionPerformed(ActionEvent e) {

	}

	public void paintComponent(Graphics g) {
		super.paintComponent(g);

	}
}
```

delay, in milliseconds, between ActionEvents

```
public class MyJPanel extends JPanel implements ActionListener {

	private Timer t;

	public MyJPanel() {

		t = new Timer(1000, this);
		t.start();
	}

	public void actionPerformed(ActionEvent e) {

	}

	public void paintComponent(Graphics g) {
		super.paintComponent(g);

	}
}
```

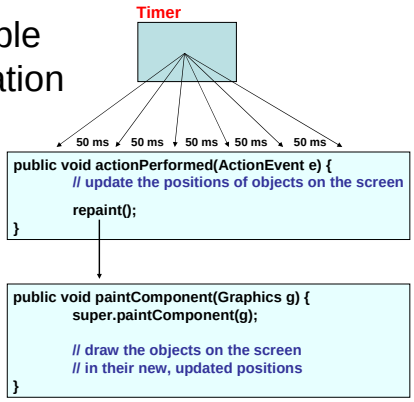
```
public class MyJPanel extends JPanel implements ActionListener {

	private Timer t;

	public MyJPanel() {

		t = new Timer(1000, this);
		t.start();
	}

	public void actionPerformed(ActionEvent e) {
		System.out.println("TICK");
	}

	public void paintComponent(Graphics g) {
		super.paintComponent(g);

	}
}
```

```
public class MyJPanel extends JPanel implements ActionListener {

	private Timer t;

	public MyJPanel() {

		t = new Timer(1000, this);
		t.start();
	}

	public void actionPerformed(ActionEvent e) {
		System.out.println("TICK");
	}

	public void paintComponent(Graphics g) {
		super.paintComponent(g);

	}
}
```

Other Timer methods

- t.stop();
- t.isRunning();
- t.setDelay(500);
- int delay = t.getDelay();

Simple animation



Example: Swarm

An array of Point objects is used to store the positions of the dots


```
public class MyJPanel extends JPanel implements ActionListener{

	private Point[] points;
	private Timer t;

	public MyJPanel() {

		addKeyListener(this);
	}

	public void actionPerformed(ActionEvent e) {

	}

	public void paintComponent(Graphics g){
		super.paintComponent(g);


	}
}
```

Initialise the instance variables

Update the position of each point in the array

Draw the points

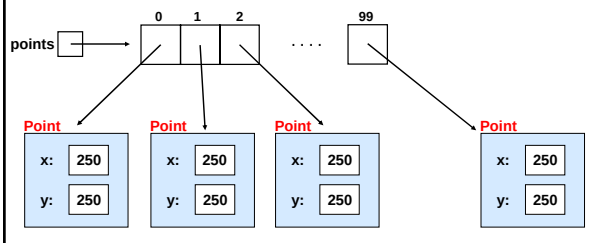
Example: Swarm

// Initialise the instance variables

```
public MyJPanel() {

	points = new Point[100];

	for (int i = 0; i < points.length; i++) {
		points[i] = new Point(250, 250);
	}

	t = new Timer(30, this);
	t.start();
}
```

Example: Swarm

- We can visualise this array of Points as below:



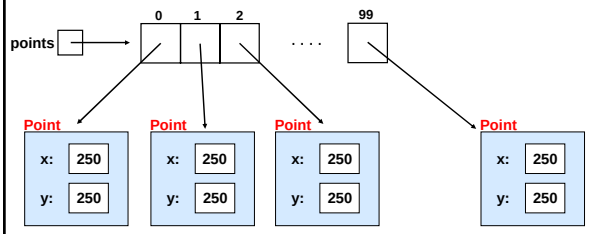
Example: Swarm

// Update the position of each point in the array

```
public void actionPerformed(ActionEvent e) {

      for (int i = 0; i < points.length; i++) {
            int dX = (int)(Math.random() * 5) - 2;
            int dY = (int)(Math.random() * 5) - 2;
            points[i].translate(dX, dY);
      }

      repaint();
}
```

Example: Swarm

- Each point is moved a random distance in the x and a random distance in the y direction:



Example: Swarm

// Draw the points

```
public void paintComponent(Graphics g) {
      super.paintComponent(g);

      for (int i = 0; i < points.length; i++) {
            g.fillOval(points[i].x-4, points[i].y-4, 8, 8);
      }
}
```