GRAU D'ENGINYERIA INFORMÀTICA

PROGRAMACIÓ II

Bloc 3:

Programació Orientada a Events (5)

Sergio Sayago (basat en material de Laura Igual)
Departament de Matemàtiques i Informàtica
Facultat de Matemàtiques i Informàtica
Universitat de Barcelona

Índex Bloc 3:
Programació Orientada a Events

- Mecanismes d'interacció
  - Interacció mitjançant flux seqüencial
  - Interacció mitjançant programació orientada a events
- Programació d'Interfícies Gràfiques d'Usuari
- Model de gestió d'events: Exemple d'implementació d'una finestra.
- Events i Listeners
- Components i Contenidors
- Classes adapter i classes internes: Exemple d'implementació d'una finestra que es tanca.
- Mes sobre swing components: Exemples
- Layout manager
- Look and feel
- **Panells i gràfics**
- Animacions

Panells i Gràfics

- Hem treballat amb panells utilitzant-los com a contenidors d'altres components. D'aquesta forma podíem establir el diagrama de la interfície definint el diagrama de cada panell.
- Un panell es pot utilitzar també per a dibuixar imatges
- Veurem llavors cóm utilitzar un objecte de la classe JPanel com un àrea dedicada específicament al dibuix.
- L'usuari podrà dibuixar amb el ratolí o podrà dibuixar determinats objectes gràfics predeterminats de les llibreries gràfiques.

Gràfics

1. Posar components en una finestra:

   frame.getContentPane().add(myBoto);

2. Dibuixar un gràfic sobre un component

   Graphics.fillOval(70,70,100,100);

3. Posar una imatge (per exemple, un jpeg)en una component:

   Graphics.drawImage(myPic,10,10,this);

Exercici 1: GUI simple

```
import java.awt.Color;
import java.awt.Graphics;
import javax.swing.JFrame;
import javax.swing.JPanel;

public class GUIPanelDeDibuix {
  public static void main (String[] args) {
     GUIPanelDeDibuix gui = new GUIPanelDeDibuix();
     gui.go();
  }

  public void go() {
     JFrame frame = new JFrame();
     frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
     PanelDeDibuix drawPanel = new PanelDeDibuix();
     frame.getContentPane().add(drawPanel);
     frame.setSize(300,300);
     frame.setVisible(true);
  }// tanca mètode go()
}// tanca classe GUIPanelDeDibuix
```

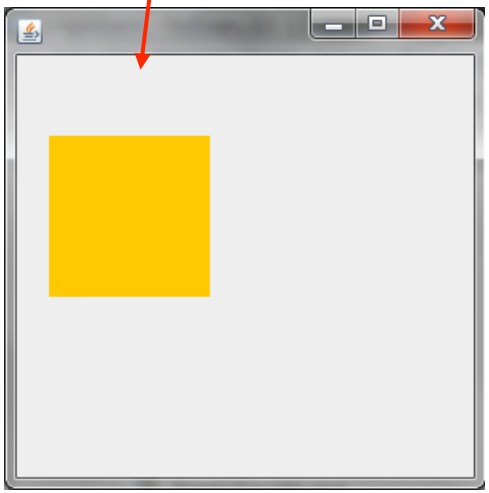
```
class PanelDeDibuix extends JPanel {

   public void paintComponent(Graphics g) {
      g.setColor(Color.orange);
      g.fillRect(20,50,100,100);
   }
}
```

Amplada i alçada.

Posició cantonada esquerra dalt

paintComponent



Mètode paintComponent()

- És un mètode **heredat** (de JComponent)
  - public class JPanel extends JComponent

- Mai cridar al mètode paintComponent() directament.

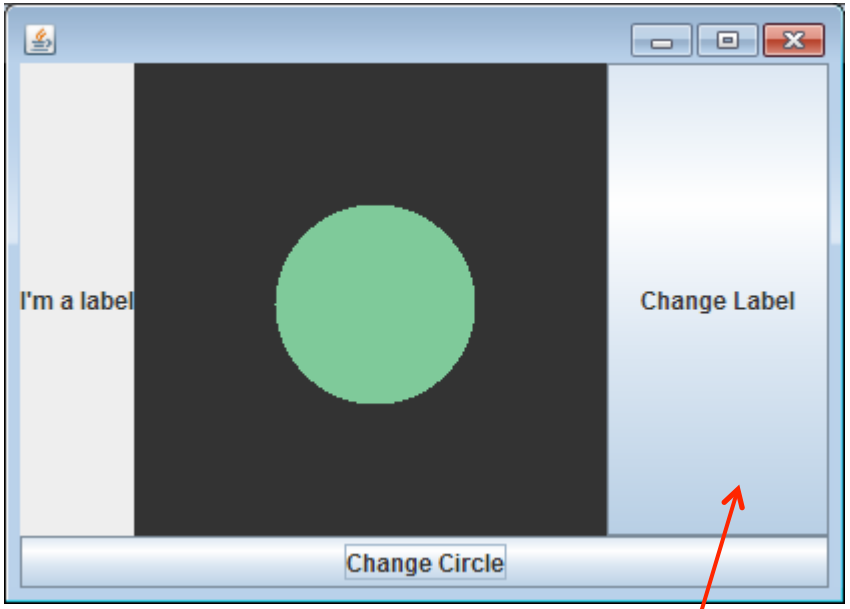
- Qui crida **paintComponent()?**
  - Es crida automàticament quan es fa visible la component o quan es canvia la seva mida.
  - Es crida indirectament des del listener definit per l'usuari via repaint()
    - Canvi de les variables de la instància.
    - Crida a repaint().

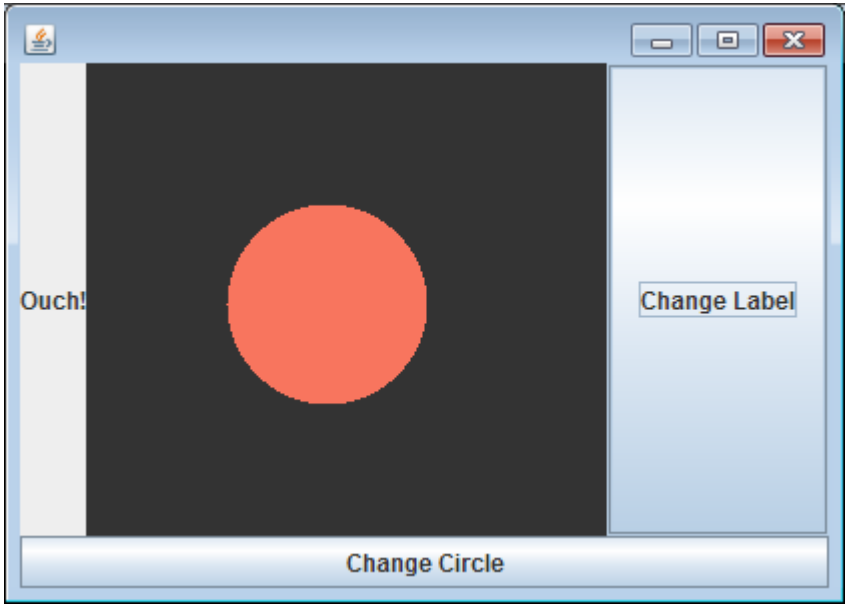
Component

- Mètode repaint()
- public void repaint() -> pinta tota la component
- public void repaint(long tm) -> pinta tota la component en el temps indicat
- public void repaint(int x, int y, int width, int height) -> pinta el rectangle indicat de la component
- public void repaint(long tm, int x, int y, int width, int height)

Exercici 2

- Implementeu una interfície gràfica d'usuari que contingui dos botons. Amb un canviem el color del cercle i amb l'altre canviem l'etiqueta. Utilitza classes internes amb nom per implementar els escoltadors.





Aquest botó canvia el color del cercle

Aquest botó canvia el text de l'altra part de la finestra

Exercici 2

Classes internes

```
import javax.swing.*;
import java.awt.*;
import java.awt.event.*;
public class TwoButtons  {
   JFrame frame;
   JLabel label;
   public static void main (String[] args) {
      TwoButtons gui = new TwoButtons();
      gui.go();
   }
   public void go() {
      frame = new JFrame();
      frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);

      JButton labelButton = new JButton("Change Label");
      labelButton.addActionListener(new LabelButtonListener());

      JButton colorButton = new JButton("Change Circle");
      colorButton.addActionListener(new ColorButtonListener());

      label = new JLabel("I'm a label");
      PanelDeDibuix drawPanel = new PanelDeDibuix ();
      frame.getContentPane().add(BorderLayout.SOUTH, colorButton);
      frame.getContentPane().add(BorderLayout.CENTER, drawPanel);
      frame.getContentPane().add(BorderLayout.EAST, labelButton);
      frame.getContentPane().add(BorderLayout.WEST, label);

      frame.setSize(420,300);
      frame.setVisible(true);
   }
   class LabelButtonListener implements ActionListener{
      public void actionPerformed(ActionEvent event) {
         label.setText("Ouch!");
      }
   }
   class ColorButtonListener implements ActionListener{
      public void actionPerformed(ActionEvent event) {
         frame.repaint();
      }
   }
} // Fi class TwoButtons
```

Mètode main

Classe definida en la següent pàgina

Crida al mètode paintComponent

Exercici 2

…. Classes internes

```
class LabelButtonListener implements ActionListener{
  public void actionPerformed(ActionEvent event) {
     label.setText("Ouch!");
  }
}
class ColorButtonListener implements ActionListener{
  public void actionPerformed(ActionEvent event) {
     frame.repaint();
  }
}
} // Fi class TwoButtons
```

Crida al mètode paintComponent

paintComponent és un mètode de Jcomponent que és sobreescrit en la subclasse de JPanel.

Refresca el panell de dibuix

```
class PanelDeDibuix extends JPanel {
 @Override
  public void paintComponent(Graphics g) {
      g.fillRect(0,0,this.getWidth(), this.getHeight());
      // make random colors to fill with
      int red = (int) (Math.random() * 255);
      int green = (int) (Math.random() * 255);
      int blue = (int) (Math.random() * 255);

      Color randomColor = new Color(red, green, blue);
      g.setColor(randomColor);
      g.fillOval(70,70,100,100);
    }
} // Fi class PanelDeDibuix
```

Els següents gràfics es pintaran amb el color aleatori definit

Índex Bloc 3:
Programació Orientada a Events

- Mecanismes d'interacció
  - Interacció mitjançant flux seqüencial
  - Interacció mitjançant programació orientada a events
- Programació d'Interfícies Gràfiques d'Usuari
- Model de gestió d'events: Exemple d'implementació d'una finestra.
- Events i Listeners
- Components i Contenidors
- Classes adapter i classes internes: Exemple d'implementació d'una finestra que es tanca.
- Mes sobre swing components: Exemples
- Layout manager
- Look and feel
- Panells i gràfics
- **Animacions**

Exercici 3: Animació simple

- Implementeu una interfície gràfica que dibuixa sobre una finestra de tamany 300x300 pixels un cercle verd a la posició inicial (70,70) i el va movent en diagonal fins arribar a la cantonada inferior dreta.

Exercici 3: Animació simple

```
import javax.swing.*;
import java.awt.*;
public class SimpleAnimation {
   int x = 70;
   int y = 70;
   public static void main (String[] args) {
      SimpleAnimation gui = new SimpleAnimation ();
      gui.go();
   }
   public void go() {
      JFrame frame = new JFrame();
      frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
      PanelDeDibuix drawPanel = new PanelDeDibuix();
      frame.getContentPane().add(drawPanel);
      frame.setSize(300,300);
      frame.setVisible(true);
      for (int i = 0; i < 130; i++) {
         x++;
         y++;
         frame.repaint();
         try {
           Thread.sleep(5);
         } catch(Exception ex) { }
      }
   }// close go() method
```

Mètode main

L'acció està aquí:

Es repeteix 130 vegades

Li diu al panel que es torni a pintar

Espera 5 milisegons

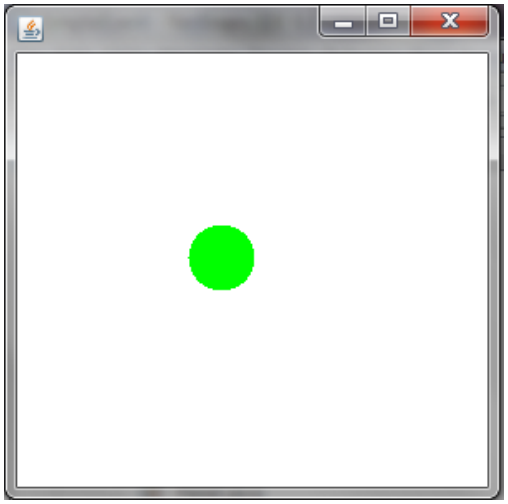
```
class PanelDeDibuix extends JPanel {
   @Override
   public void paintComponent(Graphics g) {
      g.setColor(Color.white);
      g.fillRect(0,0,this.getWidth(), this.getHeight());

      g.setColor(Color.green);
      g.fillOval(x,y,40,40);

   }
  } // close inner class
} // close outer class
```

El blanc és el primer color seleccionat

El verd és el segon



Exercici 3: Animació simple

- Com es podria fer amb classes externes? Seria millor amb classes externes?
- Què passaria si no fem aquests dues línies?

```
class PanelDeDibuix extends JPanel {
  @Override
  public void paintComponent(Graphics g) {
    g.setColor(Color.white);
    g.fillRect(0,0,this.getWidth(), this.getHeight());

    g.setColor(Color.green);
    g.fillOval(x,y,40,40);

  }
 } // close inner class
} // close outer class
```

Exercici 4: Animació simple

- Implementeu una interfície gràfica que dins d'una finestra de tamany 500x300 pixels dibuixa un rectangle blau d'amplada 500 i alçada 250 i el va fent més petit fins que desapareix.

Exercici 4: Animació simple

```
import javax.swing.*;
import java.awt.*;
public class Animate {
   private static final int XSIZE = 500;
   private static final int YSIZE = 300;
   int x = 1;
   int y = 1;
```

Mètode main

```
  public static void main (String[] args) {
     Animate gui = new Animate ();
     gui.go();
  }
  public void go() {
     JFrame frame = new JFrame();
     frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
     PanelDeDibuix panelDibuix = new PanelDeDibuix();
     frame.getContentPane().add(panelDibuix);
     frame.setSize(XSIZE,YSIZE);
     frame.setVisible(true);
     for (int i = 0; i < 125; i++) {

                                   ...

      }
   }// close go() method
```

El rectangle és dues vegades més ample que alt

```
class PanelDeDibuix extends JPanel {
}


...

  }// Fi classe PanelDeDibuix
} // Fi classe Animate
```

Exercici 4: Animació simple

```
import javax.swing.*;
import java.awt.*;
public class Animate {
   private static final int XSIZE = 500;
   private static final int YSIZE = 300;
   int x = 1;
   int y = 1;
```

Mètode main

```
   public static void main (String[] args) {
      Animate gui = new Animate ();
      gui.go();
   }
   public void go() {
      JFrame frame = new JFrame();
      frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
      PanelDeDibuix panelDibuix = new PanelDeDibuix();
      frame.getContentPane().add(panelDibuix);
      frame.setSize(XSIZE,YSIZE);
      frame.setVisible(true);
      for (int i = 0; i < 125; i++) {
        frame.repaint();
        try {
          Thread.sleep(50);
        } catch(Exception ex) { }
        x=x+2;
        y++;
      }
}// close go() method
```

El rectangle és dues vegades més ample que alt

```
class PanelDeDibuix extends JPanel {
      public void paintComponent(Graphics g ) {
            g.setColor(Color.white);
            g.fillRect(0,0,XSIZE,YSIZE);

            g.setColor(Color.blue);
            g.fillRect(x,y,500-x*2,250-y*2);
      }
   }// Fi classe PanelDeDibuix
} // Fi classe Animate
```

Exercici 5: animació amb el ratolí

- Implementeu una interfície gràfica de mida 400x200 píxels a on quan es fa clic amb el ratolí sobre ella apareixen al costat els valors de les coordenades de la posició del ratolí en aquest moment.

- Utilitza MouseAdapter per implementar l'escoltador del ratolí

RatonSwing Animation

```
public class RatonSwing{
   int x,y;
   JFrame frame;
   PanelDeDibujo panelDeDibujo;
```

Mètode main

```
   public static void main (String[] args) {
      RatonSwing rat = new RatonSwing();
      rat.go();
   }

   private void go(){
      frame = new JFrame("Jugando con el ratón ...");
      frame.addWindowListener(new GestorFinestra());

      panelDeDibujo = new PanelDeDibujo();
      frame.getContentPane().add(panelDeDibujo,BorderLayout.CENTER);
      frame.addMouseListener(new GestorRatoli());

      frame.setBounds(250,200,400,200);
      frame.setVisible(true);
   }
```

Continua en la següent pàgina….

```
    class GestorRatoli extends MouseAdapter{
        @Override
        public void mousePressed(MouseEvent e) {
            x = e.getX();
            y = e.getY();
            frame.repaint();
        }
    }
    class PanelDeDibujo extends JPanel {
         PanelDeDibujo(){
            setBackground(Color.white);
         }
        @Override
        public void paintComponent(Graphics g) {
            g.drawString(x+","+y,x,y);
        }
    }
} // Final classe RatonSwing


class GestorFinestra extends WindowAdapter{
    @Override
    public void windowClosing(WindowEvent e) {
        System.exit(0);
    }
}
```

Classe interna que hereta de la classe MouseAdapter i gestiona l'event de prémer el ratolí reescrivint el mètode mousePressed.

Crida al mètode paintComponent de la classe PanelDeDibujo.

Classe interna que hereta de Jpanel i reescriu el mètode paint

Classe que hereta de WindowAdapter i reescriu el mètode windowClosing.

Referències del bloc 3

- Alguns exemples del llibre "**Head First Java**", Kathy Sierra & Bert Bates.
- *Apunts: "***Aprenda Java como si estuviera en Primero**" (Universidad de Navarra): http://www.tecnun.es/asignaturas/Informat1/ayudainf/aprendainf/Java/Java2.pdf
- "**Creating a GUI with JFC/Swing**" (The Swing Tutorial) http://java.sun.com/docs/books/tutorial/uiswing/
- The Swing Connection http://java.sun.com/javase/technologies/desktop/articles.jsp
- Components: http://download.oracle.com/javase/tutorial/uiswing/components/