Memoria Práctica 1  
PlayStation00

GIITIN01-2-006 Comunicación Persona-Máquina

Alumno:

UO283179  
Miguel Sal Fernández

|  |  |
| --- | --- |
| Bloque | Incluido |
| 1 | Sí |
| 2 | Sí |
| 3 | Sí |
| 4 | No |
| 5 | Sí |

Contenido

[Introducción 3](#_Toc69140779)

[Temática del proyecto 3](#_Toc69140780)

[Guía de usuario básica 4](#_Toc69140781)

[Subtitulo 1 4](#_Toc69140782)

[Subtitulo 2 4](#_Toc69140783)

[Bloques 5](#_Toc69140784)

[Bloque 1. Ventana y eventos 5](#_Toc69140785)

[1.1 Barra de menús con algunos menús y submenús 5](#_Toc69140786)

[1.2 Algunos componentes en un *layout* diferente del default 5](#_Toc69140787)

[1.3 Componentes modificados por eventos 5](#_Toc69140788)

[1.4 Tratamiento de eventos de teclado 5](#_Toc69140789)

[1.5 Modificación de componentes con un número variable de elementos 6](#_Toc69140790)

[1.6 Elementos no vistos 6](#_Toc69140791)

[Bloque 2. Varias ventanas 7](#_Toc69140792)

[2.1 La ventana principal modifica componentes de otra ventana 7](#_Toc69140793)

[2.2 Una ventana no principal modifica componentes de la principal 7](#_Toc69140794)

[2.3 Una ventana no principal modifica componentes de otra ventana no principal 7](#_Toc69140795)

[2.4 Una ventana inicial que no sea la principal 8](#_Toc69140796)

[Bloque 3. Diálogos 9](#_Toc69140797)

[3.1 Un diálogo usando JOptionPane 9](#_Toc69140798)

[3.2 Un diálogo predefinido 9](#_Toc69140799)

[3.3 Un diálogo creado por el usuario que pida información al usuario 9](#_Toc69140800)

[Bloque 4. Interfaz en primer plano 10](#_Toc69140801)

[4.1 Métodos set para dar información a la tarea 10](#_Toc69140802)

[4.2 Métodos que envíen información de la tarea a la interfaz 10](#_Toc69140803)

[4.3 Posibilidad de hacer un stop 10](#_Toc69140804)

[Bloque 5. Gráficos 11](#_Toc69140805)

[5.1 Clase hija de un componente del que se refine su método Paint 11](#_Toc69140806)

[5.2 Métodos set para modificar lo que se pinta en la clase anterior 11](#_Toc69140807)

[5.3 Utilización del método repaint 11](#_Toc69140808)

[5.4 Elemento no visto en clase ni en apuntes 11](#_Toc69140809)

# Introducción

En este informe se detallará tanto el código fuente, como el funcionamiento del programa PlayStation00, el cual se presenta como proyecto de la primera parte de la asignatura Comunicación Persona-Máquina.

## Temática del proyecto

El proyecto está basado en la interfaz de una consola.

## Guía de usuario básica

El usuario puede introducir su nombre al abrir la aplicación y acceder a los juegos mostrados en la pestaña de juegos.

# Bloques

## Bloque 1. Ventana y eventos

### 1.1 Barra de menús con algunos menús y submenús

Incluir captura de pantalla. (Opcional: otros detalles).

### 1.2 Algunos componentes en un *layout* diferente del default

Indicar nombre de ventana o ventana/componente y layout que tiene.

|  |  |  |
| --- | --- | --- |
| Ventana | Componente | Layout |
| Ajustes | jLabel1 | Absolute |
| jButton1 | Absolute |
| BValorar | Absolute |
| Ventana2 |  | Layout |
| Ventana3 |  | Layout |

### 1.3 Componentes modificados por eventos

Indicar nombre de ventana, nombre de componente, clase de componente y nombre del evento (nombre del método) que modifica este componente

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Ventana | Componente | | Clase | Evento |
| Ventana Principal | **Label** | jUser | JLabel | jNombreActionPerformed |
| **Modificado** | Referencia Java |  |
| Código | | | | |
| private void jNombreActionPerformed(java.awt.event.ActionEvent evt) {  // TODO add your handling code here:  User = jNombre.getText();  jUser.setText(User);  VP.LUser1.setText(User);  } | | | | |

### 1.4 Tratamiento de eventos de teclado

Indicar el nombre de la ventana y el nombre del evento (nombre del método) que trata eventos de teclado.

|  |  |  |
| --- | --- | --- |
| Ventana | Componente | Evento |
| Inicio | Inicio.java | formKeyPressed |
| Código | | |
| private void formKeyPressed(java.awt.event.KeyEvent evt) {  // TODO add your handling code here:  if (evt.getKeyChar()==VK\_ENTER){  Vp.setVisible(true);  VInicio.setVisible(false);    } | | |

### 1.5 Modificación de componentes con un número variable de elementos

Indicar nombre de ventana, nombre de componente, clase de componente y nombre del evento (nombre del método) que modifica este componente.

|  |  |  |  |
| --- | --- | --- | --- |
| Ventana | Componente | | Evento |
| Valoracion | **jLabel1** | Label | jComboBox1ActionPerformed |
| **Modificado** | Referencia Java |
| Código | | | |
| Copiar aquí el código del evento (incluida definición de método).  private void jComboBox1ActionPerformed(java.awt.event.ActionEvent evt) {  // TODO add your handling code here:  val =jComboBox1.getSelectedItem().toString();  VAjustes.setEtiqueta(val);  } | | | |

### 1.6 Elementos no vistos

Indicar información necesaria (Ventana, Componente) para localizarlo

|  |  |  |
| --- | --- | --- |
| Ventana | Componente | Evento |
| Principal | jNombre | Metodo requestFocus() |
| Código | | |
| Copiar aquí el código del evento.  private void formWindowOpened(java.awt.event.WindowEvent evt) {  // TODO add your handling code here:  jNombre.requestFocus();  } | | |

## Bloque 2. Varias ventanas

### 2.1 La ventana principal modifica componentes de otra ventana

Nombre de la ventana de principal. Nombre de la otra ventana y nombre del componente modificado

|  |  |  |
| --- | --- | --- |
| Ventana principal | Ventana no principal | Componente |
| Principal | Pintado | LUser1 |
| Código | | |
| private void jNombreActionPerformed(java.awt.event.ActionEvent evt) {  // TODO add your handling code here:  User = jNombre.getText();  jUser.setText(User);  VP.LUser1.setText(User);  }  Copiar aquí el constructor o constructores de la ventana principal.  VP = new Pintado(); | | |

### 2.2 Una ventana no principal modifica componentes de la principal

Nombre de la ventana de principal. Nombre de la otra ventana y nombre del componente modificado

|  |  |  |
| --- | --- | --- |
| Ventana principal | Ventana no principal | Componente |
| Principal | Ajustes | jUser/jNombre |
| Código | | |
| Copiar aquí el código del evento (incluida definición de método) que modifica el componente de la ventana principal.  private void jButton1ActionPerformed(java.awt.event.ActionEvent evt) {  // TODO add your handling code here:  Vpinta.LUser1.setText(" ");  Vprin.jUser.setText(" ");  Vprin.jNombre.setText(" ");  }  Copiar aquí el constructor o constructores de la ventana no principal.  public Ajustes(Pintado Vpinta,Principal Vprin){  this();  this.Vpinta=Vpinta;  this.Vprin=Vprin;  } | | |

### 2.3 Una ventana no principal modifica componentes de otra ventana no principal

Nombre de la ventana no principal que modifica. Nombre de la otra ventana y nombre del componente modificado.

|  |  |  |
| --- | --- | --- |
| Ventana modificadora | Ventana modificada | Componente |
| Ajustes | Pintado | LUser1 |
| Código | | |
| Copiar aquí el código del evento (incluida definición de método) que modifica el componente de la otra ventana.  private void jButton1ActionPerformed(java.awt.event.ActionEvent evt) {  // TODO add your handling code here:  Vpinta.LUser1.setText(" ");  Vprin.jUser.setText(" ");  Vprin.jNombre.setText(" ");  }  Copiar aquí el constructor o constructores de la ventana no principal.  public Ajustes(Pintado Vpinta,Principal Vprin){  this();  this.Vpinta=Vpinta;  this.Vprin=Vprin;  } | | |

### 2.4 Una ventana inicial que no sea la principal

Nombre de la ventana inicial

|  |
| --- |
| Ventana |
| Inicio |
| Código |
| Copiar constructor o constructores de la ventana inicial.  public Inicio(Principal Vp){  this();  this.Vp = Vp;  this.setVisible(true);  this.VInicio = this;    } |

## Bloque 3. Diálogos

### 3.1 Un diálogo usando JOptionPane

Nombre de la ventana que tiene la llamada a este diálogo.

|  |
| --- |
| Ventana |
| Pintado |
| Código |
| Copiar el método (entero) donde se utilice un diálogo de la clase JOptionPane.  private void BFinalizarActionPerformed(java.awt.event.ActionEvent evt) {  // TODO add your handling code here:  Object[] options = { "Continuar" };  int option = JOptionPane.showOptionDialog(null, "Muy Bien!", " ", JOptionPane.DEFAULT\_OPTION,  JOptionPane.INFORMATION\_MESSAGE, null, options, options[0]);    if (option == JOptionPane.OK\_OPTION) {  VPintado.setVisible(false);  }    } |

### 3.2 Un diálogo predefinido

Nombre de la ventana que tiene la llamada a este diálogo

|  |
| --- |
| Ventana |
| Pintado |
| Código |
| Copiar el método (entero) donde se utilice un diálogo predefinido.  private void jButton2ActionPerformed(java.awt.event.ActionEvent evt) {  // TODO add your handling code here:  Graphics g = jPanel1.getGraphics();  Color c1 = JColorChooser.showDialog(null, "Seleccione un color", Color.BLACK);  g.setColor(c1);  } |

### 3.3 Un diálogo creado por el usuario que pida información al usuario

La ventana del diálogo

|  |
| --- |
| Ventana diálogo |
| NotFound |
| Código |
| Copiar el código (entero) de la clase del diálogo  import java.awt.Dimension;  import java.awt.Toolkit;  /\*  \* Click nbfs://nbhost/SystemFileSystem/Templates/Licenses/license-default.txt to change this license  \* Click nbfs://nbhost/SystemFileSystem/Templates/GUIForms/JDialog.java to edit this template  \*/  /\*\*  \*  \* @author migue  \*/  public class NotFound extends javax.swing.JDialog {  /\*\*  \* Creates new form NotFound  \*/  public NotFound(java.awt.Frame parent, boolean modal) {  super(parent, modal);  initComponents();    Dimension pantalla = Toolkit.getDefaultToolkit().getScreenSize();  int posX = (pantalla.width - this.getWidth()) / 2;  int posY = (pantalla.height - this.getHeight()) / 2;  this.setLocation(posX, posY);    }  /\*\*  \* This method is called from within the constructor to initialize the form.  \* WARNING: Do NOT modify this code. The content of this method is always  \* regenerated by the Form Editor.  \*/  @SuppressWarnings("unchecked")  // <editor-fold defaultstate="collapsed" desc="Generated Code">  private void initComponents() {  jLabel1 = new javax.swing.JLabel();  BDAceptar = new javax.swing.JButton();  jLabel1.setText("No se ha encontrado el archivo del juego");  BDAceptar.setText("Aceptar");  BDAceptar.addActionListener(new java.awt.event.ActionListener() {  public void actionPerformed(java.awt.event.ActionEvent evt) {  BDAceptarActionPerformed(evt);  }  });  javax.swing.GroupLayout layout = new javax.swing.GroupLayout(getContentPane());  getContentPane().setLayout(layout);  layout.setHorizontalGroup(  layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)  .addGroup(layout.createSequentialGroup()  .addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)  .addGroup(layout.createSequentialGroup()  .addGap(31, 31, 31)  .addComponent(jLabel1))  .addGroup(layout.createSequentialGroup()  .addGap(100, 100, 100)  .addComponent(BDAceptar)))  .addContainerGap(33, Short.MAX\_VALUE))  );  layout.setVerticalGroup(  layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)  .addGroup(layout.createSequentialGroup()  .addGap(66, 66, 66)  .addComponent(jLabel1)  .addGap(18, 18, 18)  .addComponent(BDAceptar)  .addContainerGap(47, Short.MAX\_VALUE))  );  pack();  }// </editor-fold>  private void BDAceptarActionPerformed(java.awt.event.ActionEvent evt) {  // TODO add your handling code here:  dispose();  }  /\*\*  \* @param args the command line arguments  \*/  public static void main(String args[]) {  /\* Set the Nimbus look and feel \*/  //<editor-fold defaultstate="collapsed" desc=" Look and feel setting code (optional) ">  /\* If Nimbus (introduced in Java SE 6) is not available, stay with the default look and feel.  \* For details see http://download.oracle.com/javase/tutorial/uiswing/lookandfeel/plaf.html  \*/  try {  for (javax.swing.UIManager.LookAndFeelInfo info : javax.swing.UIManager.getInstalledLookAndFeels()) {  if ("Nimbus".equals(info.getName())) {  javax.swing.UIManager.setLookAndFeel(info.getClassName());  break;  }  }  } catch (ClassNotFoundException ex) {  java.util.logging.Logger.getLogger(NotFound.class.getName()).log(java.util.logging.Level.SEVERE, null, ex);  } catch (InstantiationException ex) {  java.util.logging.Logger.getLogger(NotFound.class.getName()).log(java.util.logging.Level.SEVERE, null, ex);  } catch (IllegalAccessException ex) {  java.util.logging.Logger.getLogger(NotFound.class.getName()).log(java.util.logging.Level.SEVERE, null, ex);  } catch (javax.swing.UnsupportedLookAndFeelException ex) {  java.util.logging.Logger.getLogger(NotFound.class.getName()).log(java.util.logging.Level.SEVERE, null, ex);  }  //</editor-fold>  /\* Create and display the dialog \*/  java.awt.EventQueue.invokeLater(new Runnable() {  public void run() {  NotFound dialog = new NotFound(new javax.swing.JFrame(), true);  dialog.addWindowListener(new java.awt.event.WindowAdapter() {  @Override  public void windowClosing(java.awt.event.WindowEvent e) {  System.exit(0);  }  });  dialog.setVisible(true);  }  });  }    // Variables declaration - do not modify  javax.swing.JButton BDAceptar;  javax.swing.JLabel jLabel1;  // End of variables declaration  } |

Nombre de la ventana que tiene la llamada a este diálogo

|  |
| --- |
| Ventana |
| Principal |
| Código |
| Copiar el método (entero) donde se llama al diálogo creado por el usuario  private void BMarioKartActionPerformed(java.awt.event.ActionEvent evt) {  // TODO add your handling code here:    DNotFound.setVisible(true);  } |

## Bloque 4. Interfaz en primer plano

### 4.1 Métodos set para dar información a la tarea

Nombre de la clase donde se llaman.

|  |
| --- |
| Clase |
| Nombre Clase |
| Código |
| Copiar aquí el código completo del método que los llama. |

### 4.2 Métodos que envíen información de la tarea a la interfaz

Nombre de la clase que implementa los métodos por los que la tarea informa a la interfaz.

|  |
| --- |
| Clase |
| Nombre Clase |
| Código |
| Copiar aquí el código (completo) de estos métodos |

### 4.3 Posibilidad de hacer un stop

Copiar aquí el método que permite al usuario hacer un Stop

|  |
| --- |
| Clase |
| Nombre Clase |
| Método stop: |
| Copiar aquí el método que permite al usuario hacer un Stop |

Llamada al método stop

|  |
| --- |
| Clase |
| Nombre Clase |
| Código |
| Código completo de los métodos en los que hay una llamada a stop. |

## Bloque 5. Gráficos

### 5.1 Clase hija de un componente del que se refine su método Paint

Nombre de la clase.

|  |
| --- |
| Clase |
| LienzoDibujo |
| Código |
| Código completo del método *paint*.  public class LienzoDibujo extends JPanel{    private int xAnterior, yAnterior, xActual, yActual;  protected Graphics2D g2d;    public LienzoDibujo() {  super();  setPreferredSize(new Dimension(400, 400)); // Tamaño del panel  setBackground(Color.WHITE);  addMouseListener(new MouseAdapter() {  @Override  public void mousePressed(MouseEvent e) {  xAnterior = e.getX();  yAnterior = e.getY();  g2d = (Graphics2D) getGraphics();  }  @Override  public void mouseReleased(MouseEvent e) {  g2d.dispose();  }  });  addMouseMotionListener(new MouseMotionAdapter() {  @Override  public void mouseDragged(MouseEvent e) {  xActual = e.getX();  yActual = e.getY();  g2d.drawLine(xAnterior, yAnterior, xActual, yActual);  xAnterior = xActual;  yAnterior = yActual;  }  });    }  } |

### 5.2 Métodos set para modificar lo que se pinta en la clase anterior

Declaración de las estructuras de datos que se modifican

|  |
| --- |
| Clase |
| Nombre clase |
| Código |
| Código completo de los métodos *set*. |
| Código |
| Declaración de las estructuras de datos. |

### 5.3 Utilización del método repaint

|  |
| --- |
| Clase |
| Pintado\_BBorrarActionPerformed |
| Código |
| Código completo del método que lo llama  private void BBorrarActionPerformed(java.awt.event.ActionEvent evt) {  // TODO add your handling code here:  jPanel1.setBackground(Color.WHITE);  repaint();  } |

### 5.4 Elemento no visto en clase ni en apuntes

|  |
| --- |
| Clase |
| LienzoDibujo |
| Código |
| Indicar información necesaria (Ventana, Componente) para localizarlo  Objeto Graphics2d |