



Universidad Politécnica de Cartagena

Programación para Ingeniería Telemática.

PRÁCTICA 2. Parte 2:

Interfaz Gráfica de Usuario. Manejo básico de eventos.

Contenido

OBJETIVOS DE APRENDIZAJE.....	1
RECURSOS PROPORCIONADOS	1
EJERCICIOS A REALIZAR.....	1
EJEMPLOS BÁSICOS.....	2
MOVIMIENTO DE UN PERSONAJE UTILIZANDO EL TECLADO. CLASE JUEGO_0	3
MOVIMIENTO DE UN PERSONAJE UTILIZANDO EL TECLADO. CLASE JUEGO_1	6

Objetivos de aprendizaje

- (1) Uso extensivo de librerías gráficas de Java.
- (2) Manejadores de eventos asociados a los elementos de la interfaz de usuario. Programación dirigida por eventos e inversión de control.

Recursos proporcionados

En el aula virtual podéis encontrar el fichero comprimido `git_2014_15.zip` que contiene un proyecto Java de Eclipse con el siguiente contenido:

- Paquete `p2.basíc`: Tipos de datos básicos. Se ha añadido un nuevo tipo de datos `IView` que modela las vistas asociadas a los elementos del juego.
- Paquete `p2.model_impl`: implementaciones básicas de los tipos definidos en `p2.basíc`. A veces la implementación es parcial y se pide a los alumnos que la completen. Otras veces, los alumnos tendrán que implementar los tipos completamente.
- Paquete `p2.view_impl`: implementaciones básicas de las vistas de los objetos del juego. Se pedirá a los alumnos que implementen sus propias vistas.
- Paquete `p2.ejemplos`: En este paquete se muestran una serie de ejemplos de interfaces gráficas de usuario de complejidad creciente que pueden ser tomados como modelo por los alumnos.

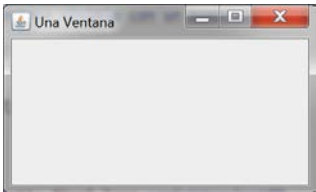
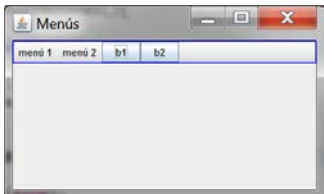


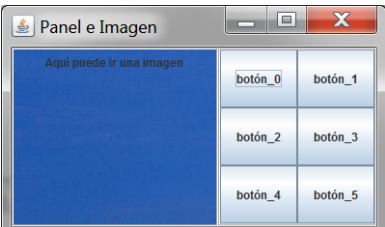

Ejercicios a realizar.


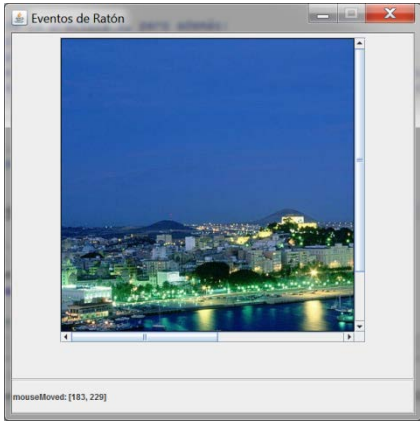
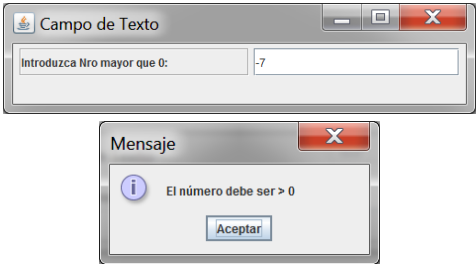
1. Modificar `Juego_0` de forma que el enlace de gusano no pueda pasar por encima de los obstáculos.
2. Modificar `Juego_0` de forma que incluya una serpiente con varios eslabones.
3. Modificar `Juego_0` de forma que al pasar por encima de una comida ésta desaparezca.
4. Modificar `Juego_0` de forma que al pasar por encima de una comida ésta desaparezca y además se sumen a la serpiente tantos puntos como valor tenga la comida.
5. Modificar `Juego_0` de forma que cada 100 ticks de reloj se le añada un eslabón (fijar el tick de reloj a 100 ms, de esta forma se añadirá un eslabón cada 10 segundos).

Antes de realizar los ejercicios lea cuidadosamente todo el boletín de prácticas.

Ejemplos básicos.

En el paquete `p2.ejemplos` se incluyen una serie de ejemplos de interfaces gráficas de usuario de complejidad creciente que pueden ser tomados como modelo por los alumnos.

<p>Ej1_UnaVentana Crea una ventana visible con un nombre.</p>	
<p>Ej2_MenusYBotones Crea una ventana visible. Añade una Barra de Menú que a su vez incluye dos menús, uno de ellos con submenús y 2 Botones.</p>	
<p>Ej2B_ConManejadoresEventos Igual que la anterior, pero con manejadores de eventos asociados a las entradas de menú y a los botones.</p>	
<p>Ej3_Paneles Crea una ventana visible, añade una Barra de Menú con diversos elementos y añade dos paneles con etiquetas identificativas.</p>	
<p>Ej4_PanelConImagen Crea una ventana visible. Añade dos paneles modelados como clases internas. En uno de ellos coloca una imagen. En el otro 6 botones colocados según un GridLayout. Se utilizan diferentes gestores de diseño.</p>	
<p>Ej4B_PanelConSelectorImagen Como el anterior pero añadiendo: Un manejadores de eventos para los botones. Un diccionario de imágenes que contiene instancias de las implementaciones de <code>IView</code>. Al pulsar un botón se establece el valor de la clave para buscar en el diccionario y se recarga la imagen correspondiente.</p>	

<p>Ej5_ImagenConScroll</p> <p>Incluye un panel con una imagen y dos barras de deslizamiento.</p> <p>El panel se modela como una clase externa.</p>	
<p>Ej5B_ConManejadoresDeEventos</p> <p>Como anterior, pero además:</p> <p>Se añade una etiqueta en la parte inferior de la ventana de la aplicación.</p> <p>El Panel de la imagen maneja los eventos de ratón que se producen sobre el mismo, escribiendo en la etiqueta el nombre del método que se ejecuta en respuesta al evento de ratón y las coordenadas en las que se ha producido el evento.</p>	
<p>Ej6_CampoDeTextoYDialogo</p> <p>Tiene una etiqueta y un campo de texto.</p> <p>Maneja los eventos que se producen en el campo de texto.</p> <p>Muestra un cuadro de diálogo que informa de los errores del operador.</p>	

Movimiento de un personaje utilizando el teclado. Clase Juego_0

En el ejemplo Juego_0:

- Se crea un tablero de juego.
- Se crean varios elementos de juego (un eslabón de serpiente y varios obstáculos y comidas).
- Se crean varias vistas (representaciones gráficas de los elementos del juego) y se asocian a los elementos del juego.
- Se utilizan las flechas de teclado para mover el enlace de la serpiente.

En la figura 1 se muestra la GUI del ejemplo y en la figura 2 se explica su estructura.

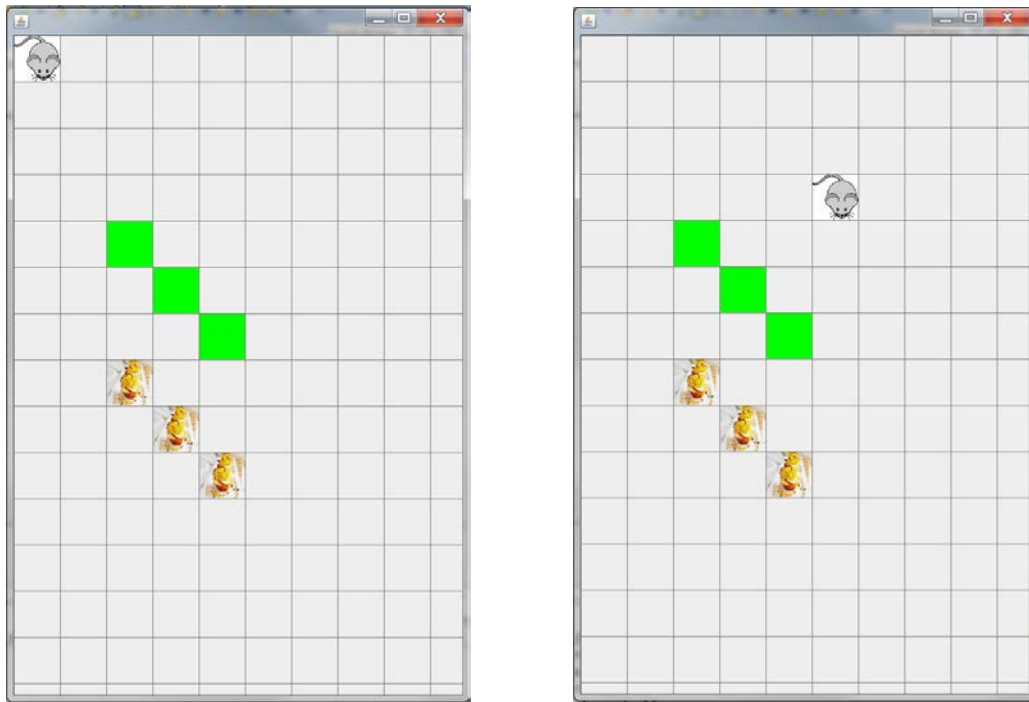


Figura 1: El personaje se mueve mediante las flechas del teclado.

```
public class Juego_0 extends JFrame implements KeyListener {

    // Paneles.
    Tablero_0 panelJuego;

    // Game object
    SnakeLink aLink;

    // Game views
    IView vObstacle, vLink, vFood;

    // View Dictionary.
    Hashtable<IGameObject, IView> tViews = new Hashtable<IGameObject, IView>();

    public Juego_0() throws IOException, JSONException, ParamException{

        panelJuego = new Tablero_0();
        getContentPane().add(panelJuego);

        addKeyListener(this);
        this.setFocusable(true);

        // Fijamos tamaño de la ventana, la hacemos visible, etc...

        // views
        vObstacle = new VSquare("obstacle");
        vFood = new VImage("food", "resources/pastel1.jpg");
        vLink = new VImage("link", "resources/raton.jpg");

        // game objects creation and association with views.
        aLink = new SnakeLink("link", "link", 1, new Coordinate(0,0));
        tViews.put(aLink, vLink);
        for (int i = 0; i < 4; i++){
            tViews.put(new Obstacle("", "", 0, new Coordinate(i+2, i+4)), vObstacle);
            tViews.put(new Fruit("", "", 0, new Coordinate(i+2, i+7)), vFood);
        }
    }
}
```

La ventana escucha eventos de teclado.

Tabla asociativa donde se guardan los objetos del juego emparejados con sus vistas. La iremos rellenando.

Creamos elementos de juego y los asociamos a las vistas.

Diferentes objetos pueden compartir la misma vista.

Figura 2: Estructura de Ejemplo_0

```

// MANEJADORES DE ENTRADAS DE TECLADO
@Override
public void keyPressed(KeyEvent arg0) { ...}

Override
public void keyReleased(KeyEvent arg0) {
    System.out.println("keyReleased: " + arg0.getKeyCode());
    int keyCode = arg0.getKeyCode();
    switch(keyCode){
        case UpKey:
            if (aLink.getCoordinate().getRow() > 0) aLink.decRow();
            break;
        case DownKey:
            if (aLink.getCoordinate().getRow() < 9) aLink.incRow();
            break;
        case LeftKey:
            if (aLink.getCoordinate().getColumn() > 0) aLink.decColumn();
            break;
        case RightKey:
            if (aLink.getCoordinate().getColumn() < 9) aLink.incColumn();
            break;
        default:break;
    }
    panelJuego.updateGame();
}

@Override
public void keyTyped(KeyEvent arg0) { ... }

// PANEL DEL TABLERO DE JUEGO.
class Tablero_0 extends JPanel {

    int lado = 60;

    public void paintComponent(Graphics g){
        super.paintComponent(g);
        for(Enumeration<IGameObject> en = tViews.keys(); en.hasMoreElements();){
            IGameObject go = en.nextElement();
            IView vi = tViews.get(go);
            vi.setSize(lado);
            vi.draw(g, go.getCoordinate().getColumn() * lado,
                    go.getCoordinate().getRow() * lado);
        }
        drawGrid(g);
    }

    public void updateGame(){
        repaint();
    }

    // Pinta cuadrícula.
    private void drawGrid(Graphics g){ }

    public static void main(String [] args) {
        Juego_0 gui = new Juego_0();
    }
}

```

La ventana escucha eventos de teclado.

Primero movemos el objeto, comprobando que no se pasa los límites del tablero.

Una vez movido el objeto, le indicamos al panel de juego que se actualice.

Clase interna que modela el tablero de juego. Accede a las variables de la clase en la que está incluida.

Figura 2 (cont.): Estructura de Juego_0

Movimiento de un personaje utilizando un temporizador. Clase Juego_1

El ejemplo `Juego_1` es prácticamente igual que `Juego_0`, pero en este caso se utiliza un temporizador para mover el enlace de la serpiente.

En la figura 3 se explica su estructura (sólo las partes claramente diferentes a `Juego_0`).

```
public class Juego_1 extends JFrame implements ActionListener {
    // Paneles.
    Tablero_1 panelJuego;

    // Game clock
    Timer timer;
    int periodo = 200;
    int ticks = 0;

    // ... Igual que Juego_0

    // View Dictionary.
    Hashtable <IGameObject, IView> tViews = new Hashtable<IGameObject, IView>();

    public Juego_1() throws IOException, JSONException, ParamException{

        // Como en Juego_0

        // Create and start timer.
        timer = new Timer(periodo, this);
        timer.start();
    }

    // ANEJADOR DE TEMPORIZADOR
    /*
    public static final int Up    = 1;
    // Other constants ...

    int currentDirection = Rigth;
    int nextCol = 0, nextRow = 0;

    public void actionPerformed(ActionEvent arg0) {

        if (currentDirection == Rigth) {
            nextCol = aLink.getCoordinate().getColumn() + 1;
            if (nextCol < 9) aLink.incColumn();
            else currentDirection = Down;
        }
        if (currentDirection == Down) {
            nextRow = aLink.getCoordinate().getRow() + 1;
            if (nextRow < 9) aLink.incRow();
            else currentDirection = Left;
        }
        if (currentDirection == Left) {
            nextCol = aLink.getCoordinate().getColumn() - 1;
            if (nextCol > 0) aLink.decColumn();
            else currentDirection = Up;
        }
        if (currentDirection == Up) {
            nextRow = aLink.getCoordinate().getRow() - 1;
            if (nextRow > 0) aLink.decRow();
            else currentDirection = Rigth;
        }
        panelJuego.updateGame();
    }
}
```

Figura 3: Estructura de `Juego_1`