

## Práctica 3 PDO | David Rodríguez Aparicio

---

En esta tercera práctica hemos añadido algunas clases nuevas y hemos adaptado el modelo-vista-controlador, como son VitaTextual, Controlador, el main, OperacionInmobiliaria y algunos enums extras.

En este caso he tenido varios problemas al ejecutar el código, los cuales son problemas de lógica, sobre todo al añadir las nuevas clases y las conexiones entre estas.

Se han solucionado algunos de estos y se solucionarían todos para dejar la práctica correcta, por ahora la vista textual se muestra pero hay pasos que no los hace correctamente.

En cuanto a la implementación de las nuevas clases, están todas completas, lo único que falta es terminar de arreglar la lógica de estos.

### VistaTextual.java

```
package juegoTexto;

/* CLASES */
import civitas.CivitasJuego;
import civitas.Diario;
import civitas.GestionInmobilarias;
import civitas.OperacionesJuego;
import civitas.Respuestas;
import civitas.SalidasCarcel;
import civitas.TituloPropiedad;
import java.util.ArrayList;
import java.util.Arrays;
import java.util.Scanner;

public class VistaTextual {

    CivitasJuego juegoModel;
    int iGestion=-1;
    int iPropiedad=-1;

    private static String separador = "=====";

    private final Scanner in;

    /* ----- Constructor ----- */

    VistaTextual () {
        in = new Scanner (System.in);
    }

    /* ----- Métodos ----- */
}
```

```
void mostrarEstado(String estado) {
    System.out.println (estado);
}

void pausa() {
    System.out.print ("Pulsa una tecla");
    in.nextLine();
}

int leeEntero (int max, String msg1, String msg2) {
    Boolean ok;
    String cadena;
    int numero = -1;
    do {
        System.out.print (msg1);
        cadena = in.nextLine();
        try {
            numero = Integer.parseInt(cadena);
            ok = true;
        } catch (NumberFormatException e) { // No se ha introducido un entero
            System.out.println (msg2);
            ok = false;
        }
        if (ok && (numero < 0 || numero >= max)) {
            System.out.println (msg2);
            ok = false;
        }
    } while (!ok);

    return numero;
}

int menu (String titulo, ArrayList<String> lista) {
    String tab = " ";
    int opcion;
    System.out.println (titulo);
    for (int i = 0; i < lista.size(); i++) {
        System.out.println (tab+i+"-"+lista.get(i));
    }

    opcion = leeEntero(lista.size(),
                       "\n"+tab+"Elige una opción: ",
                       tab+"Valor erróneo");

    return opcion;
}

SalidasCarcel salirCarcel() {
    int opcion = menu ("Elige la forma para intentar salir de la carcel",
                      new ArrayList<> (Arrays.asList("Pagando","Tirando el dado")));

    return (SalidasCarcel.values()[opcion]);
}
```

```
Respuestas comprar() {
    int opcion = menu ("Quieres comprar " +
this.juegoModel.getCasillaActual().getNombre(),
    new ArrayList<> (Arrays.asList("SI","NO"))));

    return (Respuestas.values()[opcion]);
}

void gestionar () {
    ArrayList<String> opcionesGestiones = new ArrayList<> (
        Arrays.asList(
            "VENDER",
            "HIPOTECAR",
            "CANCELAR_HIPOTECA",
            "CONSTRUIR_CASA",
            "CONSTRUIR_HOTEL",
            "TERMINAR"
        )
    );

    iGestion = menu("¿Qué gestión quiere realizar?", opcionesGestiones);

    if (GestionInmobiliarias.values()[iGestion] ==
GestionInmobiliarias.TERMINAR) {
        iPropiedad = -1;
        return;
    }

    ArrayList<String> nombresPropiedades = new ArrayList<>();

    for (TituloPropiedad propiedad :
this.juegoModel.getJugadorActual().getPropiedades()) {
        nombresPropiedades.add(propiedad.getNombre());
    }

    if (nombresPropiedades.isEmpty()) {
        System.out.println("El jugador actual no tiene propiedades que
gestionar.");
        iPropiedad = -1;
    } else {
        iPropiedad = menu(
            "¿Sobre qué propiedad quiere realizar la gestión?",
            nombresPropiedades
        );
    }
}

void mostrarSiguienteOperacion(OperacionesJuego operacion) {

    /* Esta clase no tiene toString sobrescrito ----- REVISAR -----
-- */
    System.out.println("Siguiente Operacion: " + operacion.toString());
}
```

```

    }

    void mostrarEventos() {
        while (Diario.getInstance().eventosPendientes()) {
            System.out.println("EVENTO: " + Diario.getInstance().leerEvento());
        }
    }

    void actualizarVista(){
        System.out.println (this.juegoModel.infoJugadorTexto());
    }

    public void setCivitasJuego(CivitasJuego civitas){
        this.juegoModel=civitas;
        // this.actualizarVista();
    }

    /* ----- GETTERS / SETTERS ----- */

    public int getGestion(){
        return this.iGestion;
    }

    public int getPropiedad(){
        return this.iPropiedad;
    }

}

```

## Controlador.java

```

package juegoTexto;

/* CLASES */
import civitas.CivitasJuego;
import civitas.GestionInmobilarias;
import civitas.OperacionInmobiliaria;
import civitas.OperacionesJuego;
import civitas.Respuestas;
import civitas.SalidasCarcel;

public class Controlador {
    private CivitasJuego juego;
    private VistaTextual vista;
}

```

```
/* ----- Constructor ----- */

Controlador(CivitasJuego juego, VistaTextual vista){
    this.juego = juego;
    this.vista = vista;
}

/* ----- Métodos ----- */

void juega(){

    this.vista.setCivitasJuego(this.juego);

    while(!this.juego.finalDelJuego()){
        this.vista.actualizarVista();

        this.vista.pausa();

        OperacionesJuego OperacionSiguiente = this.juego.siguientePaso();

        this.vista.mostrarSiguienteOperacion(OperacionSiguiente);

        if (OperacionSiguiente != OperacionesJuego.PASAR_TURNO) {
            vista.mostrarEventos();
        }

        if( !this.juego.finalDelJuego() ){

            switch (OperacionSiguiente) {
                case COMPRAR -> {
                    if(Respuestas.SI == this.vista.comprar()){
                        this.juego.comprar();
                    }
                }

                case GESTIONAR -> {
                    this.vista.gestionar();

                    GestionInmobiliarias gestion = GestionInmobiliarias.values()
[vista.getGestion()];
                    int ip = vista.getPropiedad();

                    OperacionInmobiliaria operacionInmobiliaria = new
OperacionInmobiliaria(gestion,ip);

                    switch (gestion) {
                        case VENDER -> {

this.juego.vender(operacionInmobiliaria.getNumPropiedad());
                        }
                    }
                }
            }
        }
    }
}
```

```
        case HIPOTECAR -> {

this.juego.hipotecar(operacionInmobiliaria.getNumPropiedad());
        }
        case CANCELAR_HIPOTECA -> {

this.juego.cancelarHipoteca(operacionInmobiliaria.getNumPropiedad());
        }

        /* Revisar estas 2 / Se deben usar desde juego y no
desde el jugador !!!!! */
        case CONSTRUIR_CASA -> {

this.juego.comprarCasa(operacionInmobiliaria.getNumPropiedad());
        }
        case CONSTRUIR_HOTEL -> {

this.juego.comprarHotel(operacionInmobiliaria.getNumPropiedad());
        }
        case TERMINAR -> {

        }
    }

    case SALIR_CARCEL -> {

        if (vista.salirCarcel() == SalidasCarcel.PAGANDO) {
            juego.salirCarcelPagando();
        } else {
            juego.salirCarcelTirando();
        }

    }

    default -> {
    }
}

this.juego.siguientePasoCompletado(OperacionSiguiente);

// System.out.println("\n--- RANKING FINAL ---");
// System.out.println(juego.ranking());

    }
}
}
```

## Main.java

```
package juegoTexto;

import civitas.CivitasJuego;
import civitas.Dado;
import java.util.ArrayList;

public class Main{

    public static void main(String[] args) {

        ArrayList<String> nombresPorDefecto = new ArrayList<>();
        nombresPorDefecto.add("David");
        nombresPorDefecto.add("Jose");
        nombresPorDefecto.add("Patri");

        Dado.getInstance().setDebug(true);

        CivitasJuego juego = new CivitasJuego(nombresPorDefecto);

        VistaTextual vista = new VistaTextual();
        Controlador controlador = new Controlador(juego, vista);

        controlador.juega();
    }

}
```

Aqui vemos un ejemplo simple de ejecución

```
PROBLEMAS 127 SALIDA CONSOLA DE DEPURACIÓN TERMINAL PUERTOS
PS C:\Users\david\Desktop\Ingenieria Informatica 2> c::; cd 'c:\Users\david\Desktop\Ingenieria Informatica 2';
-enable-preview' '-XX:+ShowCodeDetailsInExceptionMessages' '-cp' 'C:\Users\david\AppData\Roaming\Code\User\work
dhat.java\jdt_ws\Ingenieria Informatica 2_590fb48d\bin' 'juegoTexto.Main'
Casilla actual: 0
Saldo: 7500.0
Puede comprar: false
Salvoconducto: NO
Propiedades:
Pulsa una tecla
Siguiente Operacion: AVANZAR
EVENTO: Modo debug del Dado activado.
EVENTO: El jugador David se mueve a la casilla 1
EVENTO: El jugador David ha caído en la casilla: Casilla{nombre='calle1', tipo=CALLE, titulo=calle1}
Jugador: David
Estado: LIBRE
Casilla actual: 1
Saldo: 7500.0
Puede comprar: true
Salvoconducto: NO
Propiedades:
Pulsa una tecla
Siguiente Operacion: COMPRAR
EVENTO: De: INICIO_TURN0 con AVANZAR sale: DESPUES_AVANZAR
Quieres comprar Salida
0-SI
1-NO

Elige una opción: 0
Jugador: David
Estado: LIBRE
Casilla actual: 1
Saldo: 7000.0
Puede comprar: false
Salvoconducto: NO
Propiedades:
- calle1 (Casas: 0, Hoteles: 0)
```