DOCUMENTACIÓ

David Marsal Espuny
2n DAM
Curs 2020

DISSENY GRAFIC I FUNCIONAMENT DEL PROGRAMA

Aquest es el disseny grafic del programa, tenim 2 taules i cada taula te els seus camps, pots fer una insercio a la taula omplin els camps amb dades correctes si son dades incorrectes el programa t'avisara que estas introduint malament les dades i una vegada les dades siguin correctes al apretar al + s'afegira la insercio a la taula, si cliques a una fila de la taula i despres cliques a la X s'eliminara la fila, si cliques a una fila s'ompliran els camps automaticament i si canvies una dada d'un camp i cliques al icona de modificar es modificara la fila.

Tambe hi ha un filtre a la taula equips que si tu cliques i et deixara filtrar de puntuacio de menor a major o ordenar de alfabeticament els equips. A la taula jugadors pots ordenar alfabeticament els jugadors o pots ordenar gols de menor a major.

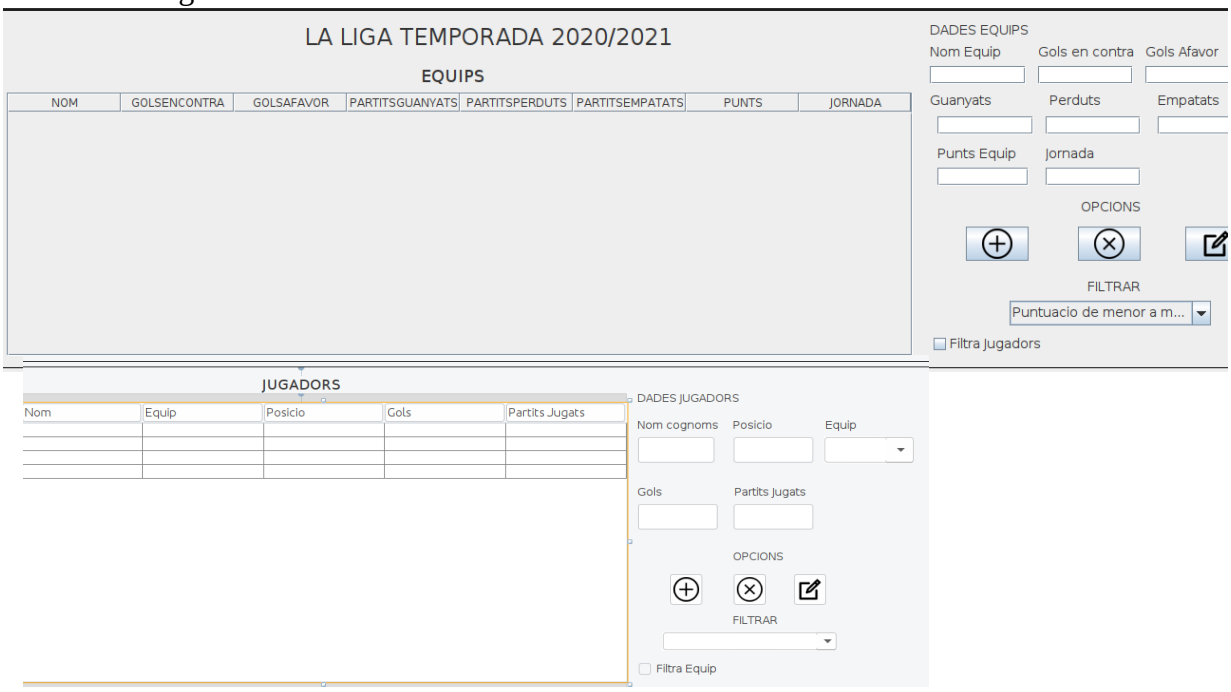
Si cliques a filtra jugadors i despres cliques a una fila de la taula equips et filtrara automaticament la taula jugadors i solament surtiran els jugadors d'aquell equip, quant desmarquis el filtre jugadors automaticament t'apareixeran tots els jugadors.

Si cliques a filtra equip i despres cliques a una fila de taula jugadors automaticament a la taula d'equips solament t'apareixera l'equip d'aquell jugador.

Els camps tenen restriccions per a que no es puguin introduir dades erronies:

A la taula equips el nom d'un equip no pot tenir numeros i per suposat no pot estar buit, la resta de camps de la taula equips son numeros per tant no et deixara introduir algo que no sigui un numero i no et deixara que deixis un camp buit i no pot ser un numero negatiu.

A la taula jugadors el nom d'un jugador no pot tenir numero i per suposat no pot estar buit, el camp posicio te de ser delanter o porter o defensa o mitg camp si introdueixes algo que sigui diferent d'aixo el programa no et deixara, el camp equip s'emplena sol ja que et deixara triar els equips que hi han a la taula equips per tant no es un camp conflictiu, als camps gols i partits jugats si introdueixes algo diferent a un numero no et deixara i tampoc et deixara que estigui buit i no pot ser un numero negatiu.



MODEL

FITXER EQUIP

CAMPS

```
 */
public class Equip implements Comparable<Equip>, Serializable {

    //private String _1_nom;
    private StringBuilder _1_nom=new StringBuilder();
    private int _2_golsEnContra;
    private int _3_golsAfavor;
    private int _4_partitsGuanyats;
    private int _5_partitsPerduts;
    private int _6_partitsEmpatats;
    private int _7_punts;
    private int _8_jornada;
    public Collection<Jugador> _9_jug = new TreeSet<>();
```

CONSTRUCTOR

```
public Equip(StringBuilder _1_nom, int _2_golsEnContra, int _3_golsAfavor, int _4_partitsGuanyats, int _5_partitsPerduts, int
    this._1_nom = _1_nom;
    this._2_golsEnContra = _2_golsEnContra;
    this._3_golsAfavor = _3_golsAfavor;
    this._4_partitsGuanyats = _4_partitsGuanyats;
    this._5_partitsPerduts = _5_partitsPerduts;
    this._6_partitsEmpatats = _6_partitsEmpatats;
    this._7_punts = _7_punts;
    this._8_jornada = _8_jornada;
}
```

GETTERS I SETTERS

```
public StringBuilder get1_nom() {
    return _1_nom;
}

public void set1_nom(StringBuilder _1_nom) {
    this._1_nom = _1_nom;
}

public int get2_golsEnContra() {
    return _2_golsEnContra;
}

public void set2_golsEnContra(int _2_golsEnContra) {
    this._2_golsEnContra = _2_golsEnContra;
}

public int get3_golsAfavor() {
    return _3_golsAfavor;
}

public void set3_golsAfavor(int _3_golsAfavor) {
    this._3_golsAfavor = _3_golsAfavor;
}

public int get4_partitsGuanyats() {
    return _4_partitsGuanyats;
}
```

```
public void set4_partitsGuanyats(int _4_partitsGuanyats) {
    this._4_partitsGuanyats = _4_partitsGuanyats;
}

public int get5_partitsPerduts() {
    return _5_partitsPerduts;
}

public void set5_partitsPerduts(int _5_partitsPerduts) {
    this._5_partitsPerduts = _5_partitsPerduts;
}

public int get6_partitsEmpatats() {
    return _6_partitsEmpatats;
}

public void set6_partitsEmpatats(int _6_partitsEmpatats) {
    this._6_partitsEmpatats = _6_partitsEmpatats;
}

public int get7_punts() {
    return _7_punts;
}

public void set7_punts(int _7_punts) {
    this._7_punts = _7_punts;
}
```

```
public int get8_jornada() {
    return _8_jornada;
}

public void set8_jornada(int _8_jornada) {
    this._8_jornada = _8_jornada;
}

public Collection<Jugador> get9_jug() {
    return _9_jug;
}

public void set9_jug(Collection<Jugador> _9_jug) {
    this._9_jug = _9_jug;
}
```

TO STRING

```
@Override
public String toString() {
    return _1_nom.toString();
}
```

I el COMPARE TO per al filtre.

```
@Override
public int compareTo(Equip o) {
    return Comparator.comparing(Equip::get7_punts).thenComparing(Equip::get1_nom).compare(this, o);
}
```

FITXER JUGADOR
CAMPS

```
private String _1_nomcognoms;
private Equip _2_equip;
private String[] _3_posicio;
private int _4_gols;
private int _5_partits;
```

CONSTRUCTOR

```
public Jugador(String _1_nomcognoms, Equip _2_equip, String[] _3_posicio, int _4_gols, int _5_partits) {
    this._1_nomcognoms = _1_nomcognoms;
    this._2_equip = _2_equip;
    this._3_posicio = _3_posicio;
    this._4_gols = _4_gols;
    this._5_partits = _5_partits;
    this._2_equip.get9_jug().add(this);

}
```

GETTERS I SETTERS

```
public String get1_nomcognoms() {
    return _1_nomcognoms;
}

public void set1_nomcognoms(String _1_nomcognoms) {
    this._1_nomcognoms = _1_nomcognoms;
}

public Equip get2_equip() {
    return _2_equip;
}

public void set2_equip(Equip _2_equip) {
    this._2_equip = _2_equip;
}

public String get3_posicio() {
    String resultat = Arrays.toString(_3_posicio);
    return resultat.substring(1, resultat.length() - 1);
}

public void set3_posicio(String[] _3_posicio) {
    this._3_posicio = _3_posicio;
}
```

```
public int get4_gols() {
    return _4_gols;
}

public void set4_gols(int _4_gols) {
    this._4_gols = _4_gols;
}

public int get5_partits() {
    return _5_partits;
}

public void set5_partits(int _5_partits) {
    this._5_partits = _5_partits;
}
```

TO STRING

```
@Override
public String toString() {
    return _1_nomcognoms;
}
```

COMPARE TO per al filtre

```
@Override
public int compareTo(Jugador o) {
    return Comparator.comparing(Jugador::get4_gols).thenComparing(Jugador::get1_nomcognoms).compare(this, o);
}
```

FITXER MODEL

Primer de tot declaracio de coleccions en tenim 2 d'equips i 2 de jugadors això ho fem per que tenim 2 filtres així que una coleccio tindra un filtre i l'altra tindra l'altre filtre.

COLECCIONS I VARIABLES PER A FITXERS

```
private static View view;
private static Collection<Equip> dades = new TreeSet<>();
private static Collection<Equip> dades2 = new TreeSet<>(new EquipOrdenaPuntuacio());
private static Collection<Jugador> dadesJugador = new TreeSet<>();
private static Collection<Jugador> dadesJugador2 = new TreeSet<>(new JugadorOrdena());
private static String fitxerEquip = "";
private static String fitxerJugador = "";
private static File fitxer;
private static File fitxer2;
private static File fitxerConfiguracio = new File("configuracio.db");
private static File fitxerConf2 = new File("output.txt");
```

GETTERS DE LES COLECCIONS

```
public static Collection<Equip> getDades() {
    return dades;
}

public static Collection<Equip> getDades2() {
    return dades2;
}

public static Collection<Jugador> getDadesJugador() {
    return dadesJugador;
}

public static Collection<Jugador> getDadesJugador2() {
    return dadesJugador2;
}
```

DADES PER DEFECTE PER A FER PROVES

```
    public Model() {
//        StringBuilder s = new StringBuilder();
//        StringBuilder s2 = new StringBuilder();
//        StringBuilder s3 = new StringBuilder();
//        Equip a1 = new Equip(s.append("a"), 1, 2, 3, 4, 5, 6, 0);
//        Equip a2 = new Equip(s2.append("b"), 2, 1, 1, 1, 1, 1, 2);
//        Equip a3 = new Equip(s3.append("c"), 2, 1, 1, 1, 1, 1, 2);
//
//        dades.add(a1);
//        dades.add(a2);
//        dades.add(a3);
//
//        String[] a = {"delanter"};
//        String[] ab = {"defensa"};
//        dadesJugador.add(new Jugador("a", a1, a, 10, 10));
//        dadesJugador.add(new Jugador("b", a1, a, 9, 9));
//        dadesJugador.add(new Jugador("c", a2, a, 8, 8));
//        dadesJugador.add(new Jugador("d", a2, ab, 7, 7));
//        dadesJugador.add(new Jugador("e", a2, ab, 6, 6));
    }
```

METODES ESCRIUREFITXER(EQUIP) I ESCRIUREFITXERJUGADOR

Tal i com indica el seu nom son 2 metodes per escriure les dades en 2 arxius i uso el try with resources i els throws per a les excepcions.

```
public static void escriureFitxer() throws FileNotFoundException, IOException {
    try(ObjectOutputStream output = new ObjectOutputStream(new BufferedOutputStream(new FileOutputStream(fitxerEquip)))) {
        for (Equip eq : Model.getDades()) {
            output.writeObject(eq);
        }
    }
}

public static void escriureFitxerJugador() throws FileNotFoundException, IOException {
    try(ObjectOutputStream output= new ObjectOutputStream(new BufferedOutputStream(new FileOutputStream(fitxerJugador)))) {
        for (Jugador eq : Model.getDadesJugador()) {
            if (eq.get2_equip() == null) {
                output.writeObject(eq);
            }
        }
    }
}
```

METODES LLEGIRFITXER(EQUIP) I LLEGIRFITXERJUGADOR

Tal i com indica el seu nom son 2 metodes per llegir les dades dels arxius i no uso el try with resources per que a l'hora d'usar try with resources amb aquests 2 metodes ma donat problemes.

```
public static void llegirFitxer() {
    ObjectInputStream input = null;
    try {
        input = new ObjectInputStream(new BufferedInputStream(new FileInputStream(fitxerEquip)));
        Equip obj;
        Jugador obj2;
        while (input != null) {
            obj = (Equip) input.readObject();
            System.out.println(obj.toString());
            Model.<Equip>insertar(obj, Model.getDades());
            for (Jugador eq : obj._9_jug) {
                Model.<Jugador>insertar(eq, Model.getDadesJugador());
            }
        }
    }catch (FileNotFoundException ex) {
    } catch (IOException ex) {
        System.out.println("Hem acabat de llegir el fitxer...");
    } catch (ClassNotFoundException ex) {
    } catch (NullPointerException ex) {
    } finally {
        try {
            input.close();
        } catch (IOException ex) {
        } catch (NullPointerException ex) {
        }
    }
}

public static void llegirFitxerJugadors() {
    ObjectInputStream input = null;
    try {
        input = new ObjectInputStream(new BufferedInputStream(new FileInputStream(fitxerJugador)));
        Jugador obj;
        while (input != null) {
            obj = (Jugador) input.readObject();
            System.out.println(obj.toString());
            Model.<Jugador>insertar(obj, Model.getDadesJugador());
        }
    } catch (FileNotFoundException ex) {
    } catch (IOException ex) {
        System.out.println("Hem acabat de llegir el fitxer...");
    } catch (ClassNotFoundException ex) {

    } finally {
        try {
            input.close();
        } catch (IOException ex) {

        } catch (NullPointerException ex) {

        }
    }
}
```

METODES GuardarContrasenya i llegirContrasenya
Al metode de guardar guardem la contrasenya en un arxiu en un lloc aleatori del arxiu entre la posicio 2 i 101 i a la primera posicio del arxiu fiquem el numero de la linia aleatoria. Amb el metode llegir la contrasenya mirem en quina linia esta ficada la contrasenya i la lleguim.

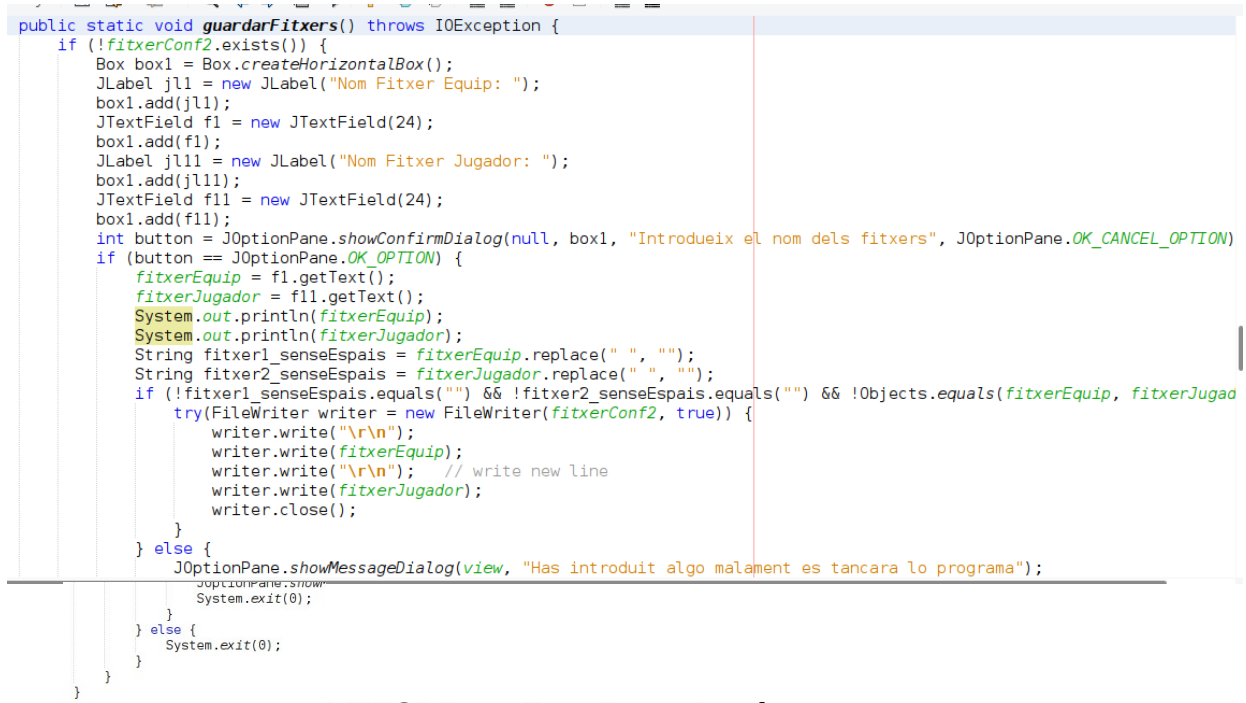
```
public static void guardarContrasenya() throws FileNotFoundException, IOException {
    Random rn = new Random();
    int offset = rn.nextInt(101 - 2);
    int contrasenya = 12345678;
    try (RandomAccessFile fitxerR = new RandomAccessFile(fitxerConfiguracio, "rw")) {
        fitxerR.seek(0);
        fitxerR.writeInt(offset);
        fitxerR.seek(offset);
        fitxerR.writeInt(contrasenya);
    }
}

public static int llegirContrasenya() throws FileNotFoundException, IOException {
    try (RandomAccessFile fitxerR1 = new RandomAccessFile(fitxerConfiguracio, "rw")) {
        fitxerR1.seek(0);
        int x = fitxerR1.readInt();
        fitxerR1.seek(x);
        return fitxerR1.readInt();
    }
}
```

METODE comprovarContrasenya
Mirem si l'arxiu on es guarda la contrasenya no existeix i si no existeix executara el metode guardarContrasenya. Una vegada estem segurs que existeix el arxiu amb la contrasenya ens mostrara un panell gracies al JoptionPane on ens demanara d'introduir la contrasenya, i si cliquem cancel·lar es tancara el programa en canvi si cliquem OK s'executara el que tenim dintre del if on ens guardem el que introdueix el usuari en una variable i també ens guardem la contrasenya correcta en una altra variable i amb un if mirem si la contrasenya que m'introdueix l'usuari i la contrasenya del fitxer es la mateixa i en el cas que sigui una contrasenya incorrecta ens donara un avis que la password es incorrecta en el cas que sigui correcta ens dira que es correcta.

```
public static void comprovarContrasenya() throws IOException {
    Box box = Box.createHorizontalBox();
    JLabel jl = new JLabel("Contrasenya: ");
    box.add(jl);
    JPasswordField jpf = new JPasswordField(24);
    box.add(jpf);
    int button = JOptionPane.showConfirmDialog(null, box, "Introdueix  la contrasenya", JOptionPane.OK_CANCEL_OPTION);
    int x = 0;
    if (!fitxerConfiguracio.exists()) {
        guardarContrasenya();
    }
    x = llegirContrasenya();
    if (button == JOptionPane.OK_OPTION) {
        char[] input = jpf.getPassword();
        String pass = String.valueOf(input);
        System.out.println(pass);

        System.out.println(String.valueOf(x));
        if (fitxerConfiguracio.exists() && pass.matches(String.valueOf(x))) {
            JOptionPane.showMessageDialog(view, "Contrasenya correcta ");
        } else {
            JOptionPane.showMessageDialog(view, "Contrasenya incorrecta ");
            System.exit(0);
        }
    } else {
        System.exit(0);
    }
}
```

METODE GuardarFitxers

Al executar el metode mirara si el fitxer on es guarda el nom dels fitxers existeix i en el cas de que no existeixi ens mostrara un JoptionPane preguntantnos quin nom volem ficarli als arxius on guardarem els equips i l'arxiu on guardarem els jugadors. En el cas que apreti CANCEL es tancara el programa i en el cas de que apreti OK s'executara lo que hi ha dintre del IF on mirem el que ens ha introduit l'usuari i ho guardem a la variable FitxerEquip el nom de l'arxiu i a la variable FitxerJugador el mateix, fem comprovacions per a que si l'usuari no introdueix res no peti, i si les dades son correctes s'escriuran en un arxiu.



```
public static void guardarFitxers() throws IOException {
    if (!fitxerConf2.exists()) {
        Box box1 = Box.createHorizontalBox();
        JLabel jl1 = new JLabel("Nom Fitxer Equip: ");
        box1.add(jl1);
        JTextField f1 = new JTextField(24);
        box1.add(f1);
        JLabel jl11 = new JLabel("Nom Fitxer Jugador: ");
        box1.add(jl11);
        JTextField f11 = new JTextField(24);
        box1.add(f11);
        int button = JOptionPane.showConfirmDialog(null, box1, "Introdueix el nom dels fitxers", JOptionPane.OK_CANCEL_OPTION)
        if (button == JOptionPane.OK_OPTION) {
            fitxerEquip = f1.getText();
            fitxerJugador = f11.getText();
            System.out.println(fitxerEquip);
            System.out.println(fitxerJugador);
            String fitxer1_senseEspais = fitxerEquip.replace(" ", "");
            String fitxer2_senseEspais = fitxerJugador.replace(" ", "");
            if (!fitxer1_senseEspais.equals("") && !fitxer2_senseEspais.equals("") && !Objects.equals(fitxerEquip, fitxerJugad
                try(FileWriter writer = new FileWriter(fitxerConf2, true)) {
                    writer.write("\r\n");
                    writer.write(fitxerEquip);
                    writer.write("\r\n");   // write new line
                    writer.write(fitxerJugador);
                    writer.close();
                }
            } else {
                JOptionPane.showMessageDialog(view, "Has introduit algo malament es tancara lo programa");
                System.exit(0);
            }
        } else {
            System.exit(0);
        }
    }
}
```

METODE nomFitxerEquip_Jugador

Mira si el fitxer on guardem el noms dels fitxers existeix i si existeix executa el que ja dintre del IF tenim un try with resources on llegeix l'arxiu i introdueix el nom dels fitxers a les seves variables corresponents.

```
public static void nomFitxerEquip_Jugador() throws FileNotFoundException, IOException {
    if (fitxerConf2.exists()) {
        try (BufferedReader br = new BufferedReader(new FileReader(fitxerConf2));) {
            String[] textData = new String[2];
            String line = br.readLine();
            for (int i = 0; i <= 1; i++) {
                textData[i] = br.readLine();
                System.out.println(textData[i]);
            }
            fitxerEquip = textData[0];
            fitxer = new File(fitxerEquip);
            fitxerJugador = textData[1];
            fitxer2 = new File(fitxerJugador);
        }
    }
}
```

METODE GENERIC INSERTAR

```
public static <T> void insertar(T eq1, Collection<T> coleccion) {
    coleccion.add(eq1);
}
```

METODES BORRAR EQUIP I BORRA JUGADOR

Apart de fer un remove de la coleccio també toca mirar els jugadors que tenen aquest equip i ficarlis el camp de jugador a null. I a l'hora de borrar un jugador també el tindrem de borrar de la coleccio de jugadors de la taula equip.

```
public static void borrarEquip(Equip eq1) {
    dades.remove(eq1);
    dades2.remove(eq1);
    for (Jugador j : eq1.get9_jug()) {
        j.set2_equip(null);
    }
}

public static void borrarJugador(Jugador j1) {
    dadesJugador.remove(j1);
    dadesJugador2.remove(j1);

    if (j1.get2_equip() != null) {
        j1.get2_equip().get9_jug().remove(j1);
    }
}
```

METODES OBTENIR EQUIP I OBTENIR JUGADOR

Com que el metode insertar es generic pues he tingut la necessitat de fer aquests dos metodes així obteneixo el equip o el jugador i seguidament uso el metode insertar per insertarlos.

```
public static void obtenirEquip(StringBuilder _1_nomEquip, int _2_golsEnContra, int _3_golsAfavor, int _4_partitsGuanyats, ir
    Equip eq1 = new Equip(
            _1_nomEquip,
            _2_golsEnContra,
            _3_golsAfavor,
            _4_partitsGuanyats,
            _5_partitsPerduts,
            _6_partitsEmpatats,
            _7_puntsEquip,
            _8_jornada
    );
    Model.insertar(eq1, dades);
    Model.insertar(eq1, dades2);
}

public static void obtenirJugador(String _1_nomcognomsJugador, Equip _2_equipJugador, String[] _3_posicioJugador, int _4_gols
    Jugador jug1 = new Jugador(
            _1_nomcognomsJugador,
            _2_equipJugador,
            _3_posicioJugador,
            _4_golsJugador,
            _5_partitsJugador
    );
    Model.insertar(jug1, dadesJugador);
    Model.insertar(jug1, dadesJugador);

}
```

METODES COMPARE per al filtre
M'ordena alfabeticament els equips i els jugadors.

```
class EquipOrdenaPuntuacio implements Comparator<Equip> {

    @Override
    public int compare(Equip o1, Equip o2) {
        return o1.get1_nom().compareTo(o2.get1_nom());
    }

}

class JugadorOrdena implements Comparator<Jugador> {

    @Override
    public int compare(Jugador o1, Jugador o2) {
        return o1.get1_nomcognoms().compareTo(o2.get1_nomcognoms());
    }
}
```

CONTROLADOR

Declaracio de variables.

```
private static Model model;
private static View view;
private TableColumn tc;
private TableColumn tc2;
private TableColumn tc3;
private int filaSel = -1;
private int filaSel2 = -1;
private int filtroEquip = 0;
private int filtroJugador = 0;
```

METODES carregarTaulaEquip i carregarTaulaJugador

Son 2 metodes que ens carreguen les taules depenent si esta seleccionat un filtre o un altre i això ho fem gracies al metode loadTable que tenim a la lliberaria, també uso el metode loadCombo per carregar el comboBox que tinc al camp equip de la taula Jugadors.

```
public void carregarTaulaEquip() {
    if (filtroEquip == 0) {
        tc = Utils.<Equip>loadTable(model.getDades(), view.getTaulaEquips(), Equip.class, true, true);
        view.getTaulaEquips().removeColumn(view.getTaulaEquips().getColumnModel().getColumn(8));
        Utils.<Equip>loadCombo(model.getDades(), view.getjComboBox1());
    } else if (filtroEquip == 1) {
        model.getDades2().addAll(model.getDades());
        tc = Utils.<Equip>loadTable(model.getDades2(), view.getTaulaEquips(), Equip.class, true, true);
        view.getTaulaEquips().removeColumn(view.getTaulaEquips().getColumnModel().getColumn(8));
        Utils.<Equip>loadCombo(model.getDades(), view.getjComboBox1());
    }
}

public void carregarTaulaJugador() {
    if (filtroJugador == 0) {
        tc2 = Utils.<Jugador>loadTable(model.getDadesJugador(), view.getTaulaJugadors(), Jugador.class, true, true);

    } else if (filtroJugador == 1) {
        model.getDadesJugador2().addAll(model.getDadesJugador());
        tc2 = Utils.<Jugador>loadTable(model.getDadesJugador2(), view.getTaulaJugadors(), Jugador.class, true, true);
    }
}
```

Al usar metodes dintre del controlador que tenen excepcions ens obliga a ficar un throws amb les possibles excepcions. Primer de tot cridem al metode comprovarContrasenya del model una vegada em aconseguit passar aquest metode voldra dir que la vista ja pot ser visible per tant fem un view.setVisible(true), seguidament ens executara el metode nomFitxerEquip_Jugador per mirar els noms dels fitxers i una vegada tenim els noms del fitxer ens lleguira el contigut dels fitxers i despres ens carregara les taules.

```
private void controlador() throws IOException, FileNotFoundException, ClassNotFoundException {

    model.comprovarContrasenya();
    view.setVisible(true);
    model.nomFitxerEquip_Jugador();
    //Combo Puntuacio
    view.getPuntuacio().addItem("Puntuacio de menor a major");
    view.getPuntuacio().addItem("Ordenar alfabeticament equips");
    view.getFiltroJugadors().addItem("Gols de menor a major");
    view.getFiltroJugadors().addItem("Ordenar alfabeticament Jugadors");

    // escriureFitxer();
    model.llegirFitxer();
    //model.escriureFitxerJugador();
    model.llegirFitxerJugadors();
    carregarTaulaJugador();
    carregarTaulaEquip();
```

AFEGIR EQUIP

Comprova que al nom de l'equip solament m'introdueix lletres, comprova que el camp no estigui buit, comprova que els numeros no son negatius i no estan buits, i si tot esta correcte m'afegeix un equip a la coleccio d'equips i carrega la Taula de nou per a que s'actualitzi automaticament la taula que veu l'usuari quant usa lo programa.

```
view.getAfegirEquip().addActionListener(
        e -> {
            Pattern pattern = null;
            pattern = Pattern.compile("^[a-zA-Z]*$");
            StringBuilder a = new StringBuilder();
            while (true) {
                String text = view.getNomEquip().getText().replace(" ", "");
                boolean found = false;
                if (text.isEmpty()) {
                    found = false;
                    JOptionPane.showMessageDialog(view, "No has introduit res, esta buit!!!");
                    break;
                }
                Matcher matcher = pattern.matcher(text);
                if (matcher.find()) {
                    try {
                        if (Integer.parseInt(view.getGolsEnContra().getText()) < 0 || Integer.parseInt(view.getGolsAfavor(
                            JOptionPane.showMessageDialog(view, "Has introduit un numero negatiu!!!");
                            found = true;
                            break;
                        }
                        model.obtenirEquip(a.append(view.getNomEquip().getText()), Integer.parseInt(view.getGolsEnContra()
                    } catch (NumberFormatException exception) {
                        JOptionPane.showMessageDialog(view, "On tenies d'introduir un numero has introduit lletres o carac
                        found = true;
                        break;
                    }
                    found = true;
                    carregarTaulaJugador();
                    carregarTaulaEquip();
                    break;
                }
                if (!found) {
                    JOptionPane.showMessageDialog(view, "No has introduit un nom de equip correcte has introduit algo mes
                    break;
                }
                break;
            }
        }
);
```

AFEGIR JUGADOR

Mira que la posicio que m'introdueix es correcta, comprova que el nom del jugador no tingui algo diferent a lletres, mira que no siguin els camps de numeros negatius, també obligara a que tots els camps estiguin plens, en el cas de que estigui tot be m'afegeix el jugador a la coleccio de jugadors.

```
view.getAfegirJugador().addActionListener(
        e -> {
            Equip obj1 = (Equip) view.getjComboBox1().getSelectedItem();
            String[] a = new String[1];
            a[0] = view.getPosicioJugador().getText();

            String[] a1 = {"Defensa"};
            String[] b = {"Delanter"};
            String[] c = {"Porter"};
            String[] d = {"Mitg camp"};
            String[] a2 = {"defensa"};
            String[] b2 = {"delanter"};
            String[] c2 = {"porter"};
            String[] d2 = {"mitg camp"};
            if (Arrays.compare(a, a1) == 0 || Arrays.compare(a, b) == 0 || Arrays.compare(a, c) == 0 || Arrays.compare(a,
                Pattern pattern = null;
                pattern = Pattern.compile("^[a-zA-Z]*$");
                while (true) {
                    String text = view.getNomJugador().getText().replace(" ", "");
                    if (text.isEmpty()) {
                        JOptionPane.showMessageDialog(view, "No has introduit res, esta buit!!!");
                        break;
                    }
                    Matcher matcher = pattern.matcher(text);
                    boolean found = false;
                    if (matcher.find()) {
                        try {
                            if (Integer.parseInt(view.getGolsJugador().getText()) < 0 || Integer.parseInt(view.getPartitsJ
                                JOptionPane.showMessageDialog(view, "Has introduit un numero negatiu!!!");
```

```
                    found = true;
                    break;
                }
                model.obtenirJugador(view.getNomJugador().getText(), obj1, a, Integer.parseInt(view.getGolsJug
            } catch (NumberFormatException exception) {
                JOptionPane.showMessageDialog(view, "On tenies d'introduir un numero has introduit lletres o c
                found = true;
                break;
            } catch (NullPointerException exception) {
                JOptionPane.showMessageDialog(view, "El jugador te de tenir un equip.");
                found = true;
                break;
            }

            found = true;
            carregarTaulaJugador();
            carregarTaulaEquip();
            break;
        }
        if (!found) {
            JOptionPane.showMessageDialog(view, "No has introduit un nom de jugador correcte has introduit alg
            break;
        }
        break;
    }
} else {
    JOptionPane.showMessageDialog(view, "No has introduit una posicio correcta te de ser Defensa o Delanter o

}
```

Al getTaulaEquips el que faig es usar el mouseClicked i així quant em clicara l'usuari en una fila automaticament m'omplira els camps, i també el que fem es mirar si el checkbox es marcat i si esta marcat solament ens mostrara els jugadors d'aquell equip a la taula jugadors.

```
view.getTaulaEquips().addMouseListener(
        new MouseAdapter() {
    @Override
    public void mouseClicked(MouseEvent e) {
        filaSel = view.getTaulaEquips().getSelectedRow();
        if (filaSel != -1) {
            DefaultTableModel model1 = (DefaultTableModel) view.getTaulaEquips().getModel();
            String nom = model1.getValueAt(filaSel, 0).toString();
            String golsEncontra = model1.getValueAt(filaSel, 1).toString();
            String golsAfavor = model1.getValueAt(filaSel, 2).toString();
            String partitsGuanyats = model1.getValueAt(filaSel, 3).toString();
            String partitsPerduts = model1.getValueAt(filaSel, 4).toString();
            String partitsEmpatats = model1.getValueAt(filaSel, 5).toString();
            String puntsEquip = model1.getValueAt(filaSel, 6).toString();
            String jornada = model1.getValueAt(filaSel, 7).toString();
            view.getNomEquip().setText(nom);
            view.getJornada().setText(jornada);
            view.getPuntsEquip().setText(puntsEquip);
            view.getPartitsEmpats().setText(partitsEmpatats);
            view.getPartitsPerduts().setText(partitsPerduts);
            view.getPartitsGuanyats().setText(partitsGuanyats);
            view.getGolsAfavor().setText(golsAfavor);
            view.getGolsEnContra().setText(golsEncontra);
        } else {
            carregarTaulaJugador();
        }
        if (view.getjCheckBox1().isSelected() == true && filaSel != -1) {
            TableColumnModel tcm = view.getTaulaEquips().getColumnModel();
            tcm.addColumn(tc);
            Equip obj = (Equip) view.getTaulaEquips().getValueAt(filaSel, tcm.getColumnCount() - 1);
            view.getNomEquip().setText(obj.toString());
            tcm.removeColumn(tc);
            carregarTaulaEquip();
            tc2 = Utils.<Jugador>loadTable(obj.get9_jug(), view.getTaulaJugadors(), Jugador.class, true, true);
        }
    }
});
```

Al getTaulaJugadors el que faig es usar el mouseClicked i així quant em clicara l'usuari en una fila automaticament m'omplira els camps, i també el que fem es mirar si el checkbox es marcat i si esta marcat solament ens mostrara el equip d'aquell jugador a la taula equips.

```
view.getTaulaJugadors().addMouseListener(new MouseAdapter() {
            @Override
            public void mouseClicked(MouseEvent e) {
                filaSel2 = view.getTaulaJugadors().getSelectedRow();
                if (filaSel2 != -1) {
                    DefaultTableModel model1 = (DefaultTableModel) view.getTaulaJugadors().getModel();
                    String nomcognoms = model1.getValueAt(view.getTaulaJugadors().getSelectedRow(), 0).toString();
                    String posicioJugador = model1.getValueAt(view.getTaulaJugadors().getSelectedRow(), 2).toString();
                    String golsJugador = model1.getValueAt(view.getTaulaJugadors().getSelectedRow(), 3).toString();
                    String partitsJugador = model1.getValueAt(view.getTaulaJugadors().getSelectedRow(), 4).toString();
                    view.getPartitsJugador().setText(partitsJugador);
                    view.getGolsJugador().setText(golsJugador);
                    view.getPosicioJugador().setText(posicioJugador);
                    view.getNomJugador().setText(nomcognoms);
                    if (view.getjCheckBox2().isSelected() == true && filaSel2 != -1) {
                        TableColumnModel tcm2 = view.getTaulaJugadors().getColumnModel();
                        tcm2.addColumn(tc2);
                        Jugador obj2 = (Jugador) view.getTaulaJugadors().getValueAt(filaSel2, tcm2.getColumnCount() - 1);
                        tcm2.removeColumn(tc2);
                        Equip eq1 = obj2.get2_equip();
                        Collection<Equip> prova = new TreeSet<>();
                        prova.add(eq1);
                        tc = Utils.<Equip>loadTable(prova, view.getTaulaEquips(), Equip.class, true, true);
                    }
                } else {
                    return;
                }
            }
```

Amb el getCheckBox1 el que faig es el mateix que amb lo getTaulaEquips però això ho faig per si l'usuari clica primer a la fila i despres al checkbox automaticament mostrara solament els jugadors d'aquell equip.

```
);

view.getjCheckBox1().addMouseListener(
        new MouseAdapter() {
    @Override
    public void mouseClicked(MouseEvent e) {
        filaSel = view.getTaulaEquips().getSelectedRow();
        if (filaSel != -1) {
            TableColumnModel tcm = view.getTaulaEquips().getColumnModel();
            tcm.addColumn(tc);
            Equip obj = (Equip) view.getTaulaEquips().getValueAt(filaSel, tcm.getColumnCount() - 1);
            tcm.removeColumn(tc);
            view.getNomEquip().setText(obj.toString());
            if (view.getjCheckBox1().isSelected() == true) {
                view.getNomEquip().setText(obj.toString());
                carregarTaulaEquip();
                tc3 = Utils.<Jugador>loadTable(obj.get9_jug(), view.getTaulaJugadors(), Jugador.class, true, true);
            } else {
                carregarTaulaEquip();
                carregarTaulaJugador();
            }
        } else {
            carregarTaulaJugador();
        }
    }
}
);
```

Amb el getCheckBox2 el fque faig es el mateix que amb lo getTaulaJugadors però això ho faig per si l'usuari clica primer a la fila i despres al checkbox automaticament mostrara solament el equip d'aquell jugador.

```
view.getjCheckBox2().addMouseListener(
        new MouseAdapter() {
    @Override
    public void mouseClicked(MouseEvent e) {
        filaSel2 = view.getTaulaJugadors().getSelectedRow();
        if (filaSel2 != -1) {
            TableColumnModel tcm2 = view.getTaulaJugadors().getColumnModel();
            tcm2.addColumn(tc2);
            Jugador obj2 = (Jugador) view.getTaulaJugadors().getValueAt(filaSel2, tcm2.getColumnCount() - 1);
            tcm2.removeColumn(tc2);
            Equip eq1 = obj2.get2_equip();
            if (view.getjCheckBox2().isSelected() == true) {
                Collection<Equip> prova = new TreeSet<>();
                prova.add(eq1);
                tc = Utils.<Equip>loadTable(prova, view.getTaulaEquips(), Equip.class, true, true);
            } else {
                carregarTaulaEquip();
                carregarTaulaJugador();
            }
        } else {
            carregarTaulaJugador();
        }
    }
}
);
```

Amb el getEliminarEquip agafem el valor del equip de la fila que selecciona l'usuari de la taula equip i usant el metode borrarEquip del model li donem el valor del equip i ens el borrara i seguidament cridem als metodes de carregar les taules.

```
view.getEliminarEquip()
        .addActionListener(
                e -> {
                    if (filaSel != -1) {
                        TableColumnModel tcm = view.getTaulaEquips().getColumnModel();
                        tcm.addColumn(tc);
                        Equip obj = (Equip) view.getTaulaEquips().getValueAt(filaSel, tcm.getColumnCount() - 1);
                        tcm.removeColumn(tc);
                        TableColumnModel tcm2 = view.getTaulaJugadors().getColumnModel();
                        tcm2.addColumn(tc2);
                        Model.borrarEquip(obj);
                        tcm2.removeColumn(tc2);
                        carregarTaulaEquip();
                        carregarTaulaJugador();
                        filaSel = -1;
                    } else {
                        JOptionPane.showMessageDialog(view, "Has de seleccionar una fila de la taula!!");
                    }
                }
        );
```

Amb el getEliminarJugador agafem el valor del jugador de la fila que selecciona l'usuari de la taula jugador i usant el metode borrarJugador del model li donem el valor del jugador i ens el borrara i seguidament cridem als metodes de carregar les taules.

```
view.getEliminarJugador()
        .addActionListener(
                e -> {
                    if (filaSel2 != -1) {
                        TableColumnModel tcm = view.getTaulaJugadors().getColumnModel();
                        tcm.addColumn(tc2);
                        Jugador obj;
                        obj = (Jugador) view.getTaulaJugadors().getValueAt(filaSel2, tcm.getColumnCount() - 1);
                        tcm.removeColumn(tc2);
                        Model.borrarJugador(obj);
                        carregarTaulaJugador();
                        carregarTaulaEquip();
                        filaSel2 = -1;
                    } else {
                        JOptionPane.showMessageDialog(view, "Has de seleccionar una fila de la taula!!");
                    }
                }
        );
```

I amb el getBotoEditar ens agafa el valor de la fila seleccionada i si canviem alguna de les dades quant li donem al boto d'editar al usar el .set ens canvia el valor del equip a la coleccio també, en aquest cas també comprovem que les dades seguint correcte igual que feiem al afegirEquip.

```
view.getBotoEditar().addActionListener(
                e -> {
                    if (filaSel != -1) {
                        TableColumnModel tcm = view.getTaulaEquips().getColumnModel();
                        tcm.addColumn(tc);
                        Equip obj = (Equip) view.getTaulaEquips().getValueAt(filaSel, tcm.getColumnCount() - 1);
                        Pattern pattern = null;
                        pattern = Pattern.compile("^[a-zA-Z]*$");
                        StringBuilder a = new StringBuilder();
                        tcm.removeColumn(tc);
                        while (true) {
                            String text = view.getNomEquip().getText().replace(" ", "");
                            if (text.isEmpty()) {
                                JOptionPane.showMessageDialog(view, "No has introduit res, esta buit!!!");
                                break;
                            }
                            Matcher matcher = pattern.matcher(text);
                            boolean found = false;
                            if (matcher.find()) {
                                try {
                                    if (Integer.parseInt(view.getGolsEnContra().getText()) < 0 || Integer.parseInt(view.g
                                        JOptionPane.showMessageDialog(view, "Has introduit un numero negatiu!!!");
                                        found = true;
                                        break;
                                    }
                                    obj.set1_nom(a.append(view.getNomEquip().getText()));
                                    obj.set2_golsEnContra(Integer.parseInt(view.getGolsEnContra().getText()));
                                    obj.set3_golsAfavor(Integer.parseInt(view.getGolsAfavor().getText()));
                                    obj.set4_partitsGuanyats(Integer.parseInt(view.getPartitsGuanyats().getText()));
```

Aqui fem el mateix que amb lo getEditarEquip però amb los jugadors.

```
view.getEditarJugador().addActionListener(
                e -> {
                    if (filaSel2 != -1) {
                        String[] a = new String[1];
                        a[0] = view.getPosicioJugador().getText();
                        String[] a1 = {"Defensa"};
                        String[] b = {"Delanter"};
                        String[] c = {"Porter"};
                        String[] d = {"Mitg camp"};
                        String[] a2 = {"defensa"};
                        String[] b2 = {"delanter"};
                        String[] c2 = {"porter"};
                        String[] d2 = {"mitg camp"};
                        if (Arrays.compare(a, a1) == 0 || Arrays.compare(a, b) == 0 || Arrays.compare(a, c) == 0 || Arrays
                            Pattern pattern = null;
                            pattern = Pattern.compile("^[a-zA-Z]*$");
                            while (true) {
                                String text = view.getNomJugador().getText().replace(" ", "");
                                if (text.isEmpty()) {
                                    JOptionPane.showMessageDialog(view, "No has introduit res, esta buit!!!");
                                    break;
                                }
                                Matcher matcher = pattern.matcher(text);
                                boolean found = false;
                                if (matcher.find()) {
                                    found = true;
                                    TableColumnModel tcm2 = view.getTaulaJugadors().getColumnModel();
                                    tcm2.addColumn(tc2);
                                    Jugador obj = (Jugador) view.getTaulaJugadors().getValueAt(filaSel2, tcm2.getColumnCou
```

```
    TableColumnModel tcm20 = view.getTaulaEquips().getColumnModel();
    tcm20.addColumn(tc);
    tcm20.removeColumn(tc);
    tcm2.removeColumn(tc2);
    try {
        Equip obj1 = (Equip) view.getjComboBox1().getSelectedItem();
        if (obj.get2_equip() == null) {
            obj.set2_equip(obj1);
            obj.get2_equip().get9_jug().add(obj);
        } else if (obj.get2_equip() != obj1) {
            obj.get2_equip().get9_jug().remove(obj);
            obj.set2_equip(null);
            obj.set2_equip(obj1);
            obj.get2_equip().get9_jug().add(obj);
        }
        String[] a3 = new String[1];
        a3[0] = view.getPosicioJugador().getText();
        obj.set3_posicio(a3);
        String nomjugador = view.getNomJugador().getText().replace(" ", "");
        obj.set1_nomcognoms(view.getNomJugador().getText());
        if (Integer.parseInt(view.getGolsJugador().getText()) < 0 || Integer.parseInt(view
            JOptionPane.showMessageDialog(view, "Has introduit un numero negatiu!!!");
            found = true;
            break;
        }
        obj.set4_gols(Integer.parseInt(view.getGolsJugador().getText()));
        obj.set5_partits(Integer.parseInt(view.getPartitsJugador().getText()));
    } catch (NumberFormatException exception) {
        JOptionPane.showMessageDialog(view, "On tenies d'introduir un numero has introduit
```

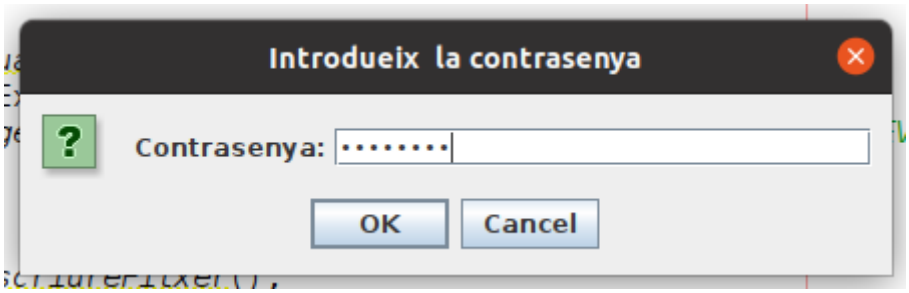
Filtres
Aqui mirem el valor del filtre i segons el que seleccioni ens carregara la taula ordenada d'una forma o d'una altra.

```
        );
    view.getPuntuacio().addItemListener(
                    e -> {
                        if (view.getPuntuacio().getSelectedIndex() == 0) {
                            filtroEquip = 0;
                            carregarTaulaEquip();
                        }
                        if (view.getPuntuacio().getSelectedIndex() == 1) {
                            filtroEquip = 1;
                            carregarTaulaEquip();
                        }
                    }
        );

    view.getFiltroJugadors().addItemListener(
                    e -> {
                        if (view.getFiltroJugadors().getSelectedIndex() == 0) {
                            filtroJugador = 0;
                            carregarTaulaJugador();
                        }
                        if (view.getFiltroJugadors().getSelectedIndex() == 1) {
                            filtroJugador = 1;
                            carregarTaulaJugador();

                        }
                    }
        );
```

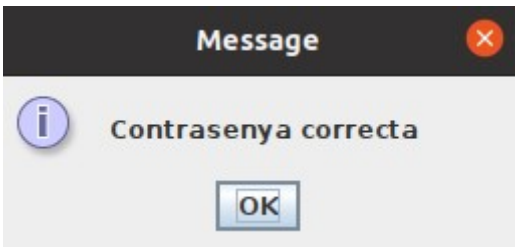
Ens surtira un JoptionPane preguntantnos si volem guardar les dades o no i si ens selecciona que no ens tancara el programa si ens selecciona que si cridara els metodes guardarFitxers que en el cas de que sigui el primer cop que l'usuari executa el programa li demanara lo nom dels fitxers i en cas de que ja tingui el nom dels fitxers no l'executara, i seguidament crida als metodes de escriureFtixer.

```
view.addWindowListener(new java.awt.event.WindowAdapter() {
    @Override
    public void windowClosing(java.awt.event.WindowEvent e) {
        int seleccion = JOptionPane.showOptionDialog(
                view, "Vol guardar les dades antes de sortir?", "Atenció!", 1, 3, null, new Object[]{"Si", "No", "Cancelar
        switch (seleccion) {
            case 0: {
                try {
                    model.guardarFitxers();
                } catch (IOException ex) {
                    Logger.getLogger(Controller.class.getName()).log(Level.SEVERE, null, ex);
                }

                try {
                    model.escriureFitxer();
                } catch (IOException ex) {
                    Logger.getLogger(Controller.class.getName()).log(Level.SEVERE, null, ex);
                }
                try {
                    model.escriureFitxerJugador();
                } catch (IOException ex) {
                    Logger.getLogger(Controller.class.getName()).log(Level.SEVERE, null, ex);
                }
                System.exit(0);
                break;
            }
            case 1:
                System.exit(0);
                break;
```

FUNCIONAMENT DE LA CONTRASENYA
Al obrir el programa ens preguntara la password, que es 12345678



Si la fiquem be ens dira contrasenya correcta.



Si la fiquem malament ens dira contrasenya incorrecta i es tancara el programa.