Disseny d’un teclat

Descripció del diagrama de classes

Grup 21.4

Versió 3.0

Jon Campillo Navarro-Soto
jon.campillo

Sergi Gonzalez Martos
sergi.gonzalez.martos

Marta Sunyer Giménez
marta.sunyer

Sandra Vázquez Rodríguez
sandra.vazquez.rodriguez

Índex

1. Diagrama del controlador i de les classes de la capa de presentació

1.1. Explicació del controlador de presentació

1.1.1. ControladorPresentacio

L'objectiu principal del controlador de presentació és la comunicació. En concret, el controlador ens serveix per a dues situacions: comunicació entre vistes i transmissió de dades entre la capa de domini i persistència amb la capa de presentació.

El “ControladorPresentacio” inclou mètodes públics que permeten la transmissió de dades entre les altres capes i la capa de presentació. Per exemple obtenir els noms dels alfabets/freqüències/textos i llistes de paraules/teclats així com afegir-los, esborrar-los o modificar-los. El controlador també actua com a comunicador entre vistes mitjançant mètodes com “tornarVistaPrincipal()”, “tornarVistaSecundaria()”, “vistaSecundariaAfegirTeclat()” o “vistaTerciariaSeleccionaDades()” entre altres.

Atributs:
- ctrlDomini: instància de controladorDomini.
- vistaPrincipal: instància de vistaPrincipal.
- vistaSecundaria: instància de VistaSecundaria.
- vistaTerciaria: instància de VistaTerciaria.

Mètodes:
+ ControladorPresentacio(): creadora.
+ inicialitzarPresentacio(): fa visible la vista principal.
+ tornarVistaPrincipal(): fa invisible la vista secundària i activa la vista principal.
+ tornarVistaSecundaria(): fa invisible la vista terciària i activa la vista secundària.
+ getNomsAlfabets(): retorna el nom dels alfabets guardats
+ getNomsDades(): retorna el nom de les dades guardades.
+ getNomsFrequencies(): retorna el nom de les freqüències de símbols guardades.
+ vistaSecundariaAfegirAlfabet(): activa la vista secundària per afegir alfabets.
+ afegeixAlfabet(nom: String, alf: ArrayList<Character>): crea un alfabet amb les dades que passen com a paràmetre.
+ esborraAlfabet(nom: String): esborra un alfabet amb el nom *nom*.
+ visualitzarAlfabet(nom: String): retorna l’alfabet amb nom *nom*
+ vistaSecundariaAfegirFrequencia(): activa la vista secundària per afegir freqüències.
+ afegeixFrequencia(nom: String, nomsDades: ArrayList<Character>): crea una freqüència de símbols amb les dades passades per paràmetre.
+ esborraFrequencia(nom: String): esborra la freqüència de símbols amb nom *nom.*
+ visualitzarFrequencia(nom: String): retorna la freqüència de símbols de nom *nom.*
+ modificaFreq(nom: String, nomNou: String): canvia el nom de la freqüència identificada per *nom* a *nomNou*.
+ afegeixLlistaDeParaules(nom: String, path: String): crea una llista de paraules amb les dades passades per paràmetre.

+ afegeixText(nom: String, path: String): crea un text amb les dades passades per paràmetre.
+ esborraDada(nom: String): esborra la dada amb nom *nom.*
+ visualitzarDada(nom: String): retorna la dada amb nom *nom.*
+ modificaDada(nom: String, nomNou: String, nouContingut: String): permet modificar el nom o el contingut d'una dada.
+ VistaSecundariaAfegirDada(r: int): activa la vista secundària per afegir una dada.
+ afegeixDadaDirectament(nom: String, contingut: String, tipus: int): permet afegir una dada donat un text que s'escriu en el moment.
+ vistaSecundariaAfegirTeclat(): activa la vista secundària per afegir un teclat.
+ afegeixTeclatDades(nomTeclat: String, nomDades: String, metode: String): crea un teclat amb les dades *nomDades* i utilitzant el mètode *metode.*
+ afegeixTeclatFrequencia(nomTeclat: String, nomFreq: String, metode: String): crea un teclat amb la freqûència de símbols *nomFreq* i utilitzant el mètode *metode.*
+ afegeixTeclatAlfFreq(nomTeclat: String, nomFreq: String, nomAlfabet: String, metode: String): crea un teclat amb la freqûència de símbols *nomFreq* i l'alfabet *nomAlfabet* i utilitzant el mètode *metode.*
+ esborraTeclat(nom: String): esborra el teclat *nom*.
+ visualitzarTeclat(nom: String): retorna el teclat *nom.*
+ modificaTeclat(nom: String, nomNou: String): permet modificar el nom d'un teclat.
+ modificaTeclat(nom: String, assignacio: char[]): permet modificar l'assignació (tecles variables del teclat principal) d'un teclat
+ getInfoTeclat(nom: String): retorna la informació del teclat nom.
+ vistaSecundariaSetDadesTeclat(nomDades: String): indica que la dada *nomDades* s'utilitzarà per crear un teclat.
+ vistaSecundariaSetAlfabetTeclat(nomAlfabet: String): indica que l'alfabet *nomAlfabet* s'utilitzarà per crear un teclat.
+ vistaSecundariaMostraPuntuacio(puntuacio: String): activa la vista secundària per mostrar una puntuació.
+ vistaPrincipalSetTextAAvaluar(text: String): passa a la vista principal el text que s'utilitza per avaluar un teclat.
+ avaluaTeclat(teclat: String, text: String): avalua un teclat donat un text.
+ vistaTerciariaSeleccionaDades(): activa la vista terciària per seleccionar dades.
+ vistaTerciariaSeleccionaTexts(): activa la vista terciària per seleccionar dades.
+ vistaTerciariaSeleccionaFrequencia(): activa la vista terciària per seleccionar freqüències de símbols.
+ vistaTerciariaSeleccionaAlfabet(): activa la vista terciària per seleccionar un alfabet.

1.2. Explicació de les classes de presentació

1.2.1. VistaPrincipal

La vista principal és l'encarregada de mostrar la pantalla d'inici amb els botons: alfabets, freqüències, textos/llistes de paraules i teclats. També és l'encarregada de mostrar la pantalla dels apartats esmentats anteriorment juntament amb els seus respectius botons d'afegir i esborrar i modificar. També és la vista utilitzada per a visualitzar el contingut d'un alfabet, freqüència, text/llista de paraules o teclat. Aquesta classe es compon de 5 aspectes principals.

1. Atributs:
    - S'utilitzen diversos components de GUI com JButton per a seleccionar les opcions de visualització.
    - JTextArea s'utilitza per a mostrar la llista d'objectes guardats.
    - DefaultListModel juntament amb JList s'usen per a gestionar i mostrar aquests objectes.
    - JPanel s'utilitza per a organitzar botons i informació rellevant.
    - JMenuItem proporciona accés a diferents parts de l'aplicació mitjançant el menú.

2. Constructor de la VistaPrincipal:
    - La classe té una constructora encarregada de la inicialització.

3. Mètodes Públics:
    - Inclou mètodes per a agregar elements a les llistes, canviar panells, i activar o desactivar el JFrame.
    - També es compten amb mètodes per a la correcta visualització dels teclats.

4. Listeners:
    - Activats en realitzar doble clic sobre un objecte guardat, criden al ControladorPresentacio per a obtenir i mostrar l'objecte.
    - Un altre tipus de listeners s'activa en fer clic en un botó d'agregar, eliminar o modificar un objecte.
    - Aquests listeners agreguen, eliminen o modifiquen l'objecte respectiu de la llista i criden al ControladorPresentacio per a dur a terme l'operació.
    - També existeixen listeners activats en fer clic en una opció del menú.

5. Mètodes Privats:
    - Inclouen inicializadores de components, FrameVista, menú, panells amb contingut, panells amb botons i informació, panells que mostren la llista d'alfabets/freqüències/textos i llistes de paraules/teclats, i panells per a visualitzar el contingut d'un objecte específic.
    - Alguns són mètodes auxiliars per la gestió de la visualització i edició de teclats.
    - A més, s'inclou una funció que obté les dades de persistència per a carregar-los en iniciar l'aplicació.

Atributs:

```
private static Set<Character> alf_cat = new HashSet<>()
{
    {
        add('\u0061'); add('\u0062');
        add('\u0063'); add('\u0064');
        add('\u0065'); add('\u0066');
        add('\u0067'); add('\u0068');
        add('\u0069'); add('\u006A');
        add('\u006B'); add('\u006C');
        add('\u006D'); add('\u006E');
        add('\u006F'); add('\u0070');
        add('\u0071'); add('\u0072');
        add('\u0073'); add('\u0074');
        add('\u0075'); add('\u0076');
        add('\u0077'); add('\u0078');
        add('\u0079'); add('\u007A');
        add('\u00E1'); add('\u00E9');
        add('\u00ED'); add('\u00F3');
        add('\u00FA'); add('\u00E0');
        add('\u00E8'); add('\u00EC');
        add('\u00F2'); add('\u00F9');
        add('\u00F1'); add('\u00E7');
    }
};
```

```
String help = "<html>L'aplicació de generació de teclats és una eina dissenyada per crear dissenys de teclats personalitzats que optimitzaran la velocitat d'escriptura.<br><br>"+
        "Escull un text o una llista de paraules i genera un teclat! És tan senzill com això.<br><br>"+
        "Si vols combinar diversos textos o llistes de paraules pots fer-ho generant una freqüència a partir d'ells.<br><br>"+
        "Finalment si a més vols afegir lletres que no surten als textos o llistes, utilitza l'alfabet desitjat al generar el teclat.<br><br>"+
        "Els teclats es generaran de forma que escriure els textos o llistes escollits sigui el més àgil possible, però si alguna cosa no t'agrada, sempre es pot editar.</html>";

static String[] sino = {"Si","No"};
String[] ok = {"OK"};

ControladorPresentacio ctrlPresentacio;

private JFrame frameVista = new JFrame("Generador de teclats");
private JPanel panelContinguts = new JPanel();
private JPanel panelInformacio = new JPanel();
private JPanel panelBotons = new JPanel();
private JPanel panelInici = new JPanel();
private VistaDialeg vistaDialegHelp = new VistaDialeg();
```

```
private JPanel panelLlistaAlfabet = new JPanel();
private JPanel panelOpcionsAlfabet = new JPanel();
private JPanel panelVisualitzarAlfabet = new JPanel();
private boolean found_s = false;

private JButton buttonAlfabet = new JButton("Alfabets");
private DefaultListModel<String> modelAlfabet = new DefaultListModel<String>();
private JList<String> listAlfabet = new JList<String>(modelAlfabet);
private JButton buttonAfegirAlfabet = new JButton("Afegir");
private JButton buttonEsborrarAlfabet = new JButton("Esborrar");
private JTextArea textAreaAlfabet = new JTextArea(5,25);
private JTextField textFieldAlfabetTitol = new JTextField();
private String actual_alf;
private String actual_contingut_alf;
private JButton buttonModificarAlfabet = new JButton("Guardar");

private JPanel panelLlistaFrequencia = new JPanel();
private JPanel panelOpcionsFrequencia = new JPanel();
private JPanel panelVisualitzarFrequencia = new JPanel();
private JPanel panelContingutFrequencia = new JPanel();

private JButton buttonFrequencia = new JButton("Freqüències");
private DefaultListModel<String> modelFrequencia = new DefaultListModel<String>();
private JList<String> listFrequencia = new JList<String>(modelFrequencia);
private JButton buttonAfegirFrequencia = new JButton("Afegir");
private JButton buttonEsborrarFrequencia = new JButton("Esborrar");
private JTextField textFieldFreq = new JTextField();
private String actual_freq;
private JButton buttonModificarFreq = new JButton("Guardar");

private JPanel panelLlistaDades = new JPanel();
private JPanel panelOpcionsDades = new JPanel();
private JPanel panelVisualitzarDades = new JPanel();
private JTextArea textAreaDades = new JTextArea(10,50);
private JTextField textFieldDades = new JTextField();
private String actual_dada;
private String actual_contingut_dada;

private JButton buttonDades = new JButton("Texts/Llistes de paraules");
private DefaultListModel<String> modelDades = new DefaultListModel<String>();
private JList<String> listDades = new JList<String>(modelDades);
private JButton buttonAfegirDades = new JButton("Afegir");
private JButton buttonEsborrarDades= new JButton("Esborrar");
private JButton buttonModificarDades = new JButton("Guardar");

private JPanel panelLlistaTeclat = new JPanel();
private JPanel panelOpcionsTeclat = new JPanel();
```

```
private JPanel panelVisualitzarTeclat = new JPanel();
private JPanel panelOpcionsVisualitzarTeclat = new JPanel();
private JPanel panelMatriuTeclat = new JPanel();

private JButton buttonTeclat = new JButton("Teclats");
private DefaultListModel<String> modelTeclat = new DefaultListModel<String>();
private JList<String> listTeclat = new JList<String>(modelTeclat);
private JButton buttonAfegirTeclat = new JButton("Afegir");
private JButton buttonEsborrarTeclat = new JButton("Esborrar");
private JButton buttonModificarTeclat = new JButton("Editar");
private JButton buttonGuardarTeclat = new JButton("Guardar");
private JButton buttonCancelarTeclat = new JButton("Cancelar");
private JButton buttonAvaluarTeclat = new JButton("Avaluar");
private JTextField textFieldTeclat = new JTextField();
private String actual_teclat;
private char[] assignacioInicialTeclat;
private char[] assignacioModificadaTeclat;

private ArrayList<JButton> caractersTeclat;
private ArrayList<JButton> especialsTeclat;
private JButton buttonCanviaTeclatSecundari = new JButton("123");
private JButton buttonCanviaTeclatPrincipal = new JButton("abc");
boolean teclatEnEdicio;
JButton teclaSeleccionada;
boolean teclatModificat;

private JPanel panel_actual = new JPanel();

private JMenuBar menubarVista = new JMenuBar();
private JMenu menuFile = new JMenu("File");
private JMenuItem menuitemQuit = new JMenuItem("Surt");
private JMenuItem menuitemHelp = new JMenuItem("Ajuda");
private JMenu menuOpcions = new JMenu("Opcions");
private JMenuItem menuitemAlfabet = new JMenuItem("Alfabets");
private JMenuItem menuitemFreq = new JMenuItem("Freqüències");
private JMenuItem menuitemDades = new JMenuItem("Texts/Llistes de paraules");
private JMenuItem menuitemTeclat = new JMenuItem("Teclats");
private JMenuItem menuitemInici = new JMenuItem("Inici");
```

Mètodes:

+ VistaPrincipal(ctrlPresentacio: ControladorPresentacio): creadora donat un controlador de presentació.
+ ferVisible(): fa visible la vista principal.
+ activar(): permet interactuar amb la vista principal.
+ desactivar(): no permet interacció amb la vista principal.
+ afegirElement(s: String, nom: String): donat un element s que pot ser alfabet, dada, freqüència o teclat, afegeix l'element a la llista corresponent.

+ canviarPanel(nou: JPanel): fa visible i permet interactuar amb els elements dins el panel nou.
+ setTextAAvaular(text: String): avalua el teclat actual amb el text passat per paràmetre.
+ visualitzarTeclatPrincipal(teclat: String): mostra el teclat principal
+ visualitzarTeclatSecundari(teclat: String): mostra el teclat secundari

- traduccioIntegerTeclat(entrada: String): tradueix l'entrada en un arrayList de Integers
- traduccioStringTeclat(entrada: String): tradueix l'entrada en un arrayList de Strings
- swap(charArray: char[], c1: char, c2: char): intercanvia les posicions dels caràcters *c1* i *c2* a l'array *charArray*.
- surtModificacioTeclat(): reinicia l'aparença del teclat a quan no s'està modificant.
- mouse_listeners(): accions que s'activen al fer doble clic sobre un element d'una llista.
- asignar_listenersComponents(): accions que s'activen al prémer un botó o un element del menú.
- inicialitzarComponents(): funció que crida a totes les altres funcions inicialitzadores.
- inicialitzar_frameVista(): inicialitza el frame.
- inicialitzar_menubarVista(): inicialitza la barra de menús.
- inicialitzar_panelContinguts(): inicialitza el panel que conté el panel de botons i el panel on es fan les accions principals.
- inicialitzar_panelBotons(): inicialitza el panel que permet viatjar a les diferents vistes a través de botons.
- inicialitzar_panelInformacio(): inicialitza el panel on es mostra la informació principal.
- inicialitzar_panelLlista(): inicialitza els diferents panels que mostren les llistes d'elements.
- inicialitzar_panelVisualitzar(): inicialitza els diferents panels que mostren el contingut d'un element.
- setTextFrequencia(freq: String): coloca la freqüència *freq* en un format adequat al panel de visualitzar freqüències.
- carregar_dadesGuardades(): carrega les dades guardades anteriorment a les llistes d'elements.

1.2.2. VistaSecundaria

La vista secundària és l'encarregada de mostrar la pantalla que permet a l'usuari modificar alfabet, freqüència, text/llista de paraules i teclat. Aquesta classe es compon de 5 aspectes principals.

1. Atributs de la Classe:
    - Es van utilitzar components de GUI com JButton per a representar els botons "Guardar" ("Generar" en el cas del teclat) i "Cancel·lar" per a les pantalles d'alfabets, freqüències i textos/llistes de paraules.
    - JTextArea s'utilitza per a mostrar els objectes guardats fins al moment.
    - DefaultListModel juntament amb JList s'utilitzen per a gestionar i mostrar aquests objectes.
    - JRadioButton s'usa per a seleccionar el tipus de dada en generar un teclat.

2. Constructor de la VistaSecundaria:
    - La classe compta amb una constructora que s'encarrega de la inicialització.

3. Mètodes Públics:

- Inclou mètodes com “afegirAlfabet()” que canvien el panell actual pel panell d'agregar alfabet.
- També trobem mètodes com `setDadesTeclat()` que permeten guardar les dades seleccionades en crear un teclat.
- Finalment, estan els mètodes que es diuen en fer clic en un botó de guardar, cancel·lar o generar. La seva funció principal és cridar al “ControladorPresentacio” per a guardar els nous objectes i tornar a la “VistaPrincipal”.

4. Listeners:

- S'utilitzen en fer clic en un botó i la seva funció és cridar als mètodes públics esmentats anteriorment.

5. Mètodes Privats:

- A més dels inicializadores de components i `frameVista`, la `VistaSecundaria` també inclou mètodes que inicialitzen els panells que apareixen en agregar un nou alfabet, freqüència, text/llesta de paraules o teclat.

Atributs:

```
private static Set<Character> alf_cat = new HashSet<>()
   {
      {
         add('\u0061'); add('\u0062');
         add('\u0063'); add('\u0064');
         add('\u0065'); add('\u0066');
         add('\u0067'); add('\u0068');
         add('\u0069'); add('\u006A');
         add('\u006B'); add('\u006C');
         add('\u006D'); add('\u006E');
         add('\u006F'); add('\u0070');
         add('\u0071'); add('\u0072');
         add('\u0073'); add('\u0074');
         add('\u0075'); add('\u0076');
         add('\u0077'); add('\u0078');
         add('\u0079'); add('\u007A');
         add('\u00E1'); add('\u00E9');
         add('\u00ED'); add('\u00F3');
         add('\u00FA'); add('\u00E0');
         add('\u00E8'); add('\u00EC');
         add('\u00F2'); add('\u00F9');
         add('\u00F1'); add('\u00E7');
      }
   };

private static String[] sino = {"Si","No"};
private static String[] error = {"D'acord"};

private ControladorPresentacio ctrlPresentacio;
```

```
private JFrame frameVista = new JFrame();

private JPanel panelContinguts = new JPanel();
private JButton buttonCancel = new JButton("Cancelar");
private JButton buttonGuardar = new JButton("Guardar");

private JPanel panel_afegirAlfabet = new JPanel();
private JTextArea textAreaInfo = new JTextArea(5,25);
private JTextField textFieldTitolAlfabet = new JTextField(25);
private boolean found_s = false;

private JPanel panel_afegirDada = new JPanel();
private JTextField textFieldTitolDada = new JTextField(25);
private JTextArea textAreaInfoDada = new JTextArea(10,25);
private JButton buttonCancelarDada = new JButton("Cancelar");
private JButton buttonGuardarDada = new JButton("Guardar");
private int tipus_dada;

private JPanel panel_afegirFrequencia = new JPanel();
private JTextField textFieldTitolFrequencia = new JTextField(25);
private JButton buttonGuardarFreq = new JButton("Guardar");
private JButton buttonCancelarFreq = new JButton("Cancelar");
private JPanel panelLlistaDades = new JPanel();
private DefaultListModel<String> modelDades = new DefaultListModel<String>();
private JList<String> listDades = new JList<String>(modelDades);

private JPanel panel_afegirTeclat = new JPanel();
private JTextField textFieldTitolTeclat = new JTextField(25);
private ButtonGroup grupTeclatDades = new ButtonGroup();
private ButtonGroup grupTeclatMetode = new ButtonGroup();
private JRadioButton rbTeclatTextLlista = new JRadioButton("Text/Llista de freqüències");
private JRadioButton rbTeclatFrequencia = new JRadioButton("Freqüència");
private JRadioButton rbTeclatQAP = new JRadioButton("<html>Branch and bound - Mètode
òptim però pot trigar molt.</html>");
private JRadioButton rbTeclatHC = new JRadioButton("<html>Cerca local - Mètode ràpid. La
seva qualitat serà més<br> alta amb més iteracions, però també augmentarà el<br> temps
de generació.</html>");
private JTextField iteracionsHC = new JTextField("20", 10);
private JButton buttonSelDades = new JButton("Seleccionar dades");
private JButton buttonSelAlfabet = new JButton("Seleccionar");

private JButton buttonGenerarTeclat = new JButton("Generar");
private JButton buttonCancelarTeclat = new JButton("Cancelar");
private String nomDadesTeclat;
private String nomAlfabetTeclat;
private String metodeTeclat;
SwingWorker<Void, Void> workerGenTeclat;
```

private JPanel panel_mostrarPuntuacio = new JPanel();
private JLabel labelPuntuacio = new JLabel();

private JPanel panel_actual = new JPanel();

Mètodes:
+ VistaSecundaria(ctrlPresentacio: ControladorPresentacio): creadora donat un ControladorPresentacio
+ ferVisible(): fa visible la vista secundaria.
+ ferInvisible(): fa invisible la vista secundaria.
+ activar(): permet interaccionar amb la vista secundaria.
+ desactivar(): no permet interacció amb la vista sescundaria.
+ afegirAlfabet(): mostra un panell on s'entren les dades per crear un alfabet.
+ afegirFrequencia(): mostra un panel on es pot seleccionar les dades de les que es vol treure una freqüència de símbols.
+ afegirTeclat(): mostra un panel on es pot escollir quins tipus de dades s'utilitzarán per crear el teclat.
+ setDadesTeclat(nomDades: String): indica que la dada *nomDades* s'utilitzarà per crear un teclat.
+ setAlfabetTeclat(nomAlfabet: String): indica que l'alfabet *nomAlfabet* s'utilitzarà per crear un teclat.
+ actionPerformed_buttonGuardar(event: ActionEvent): guarda el contingut que hi ha en els elements d'afegir Alfabet en el moment de clicar el botó Guardar.
+ actionPerformed_buttonGuardarFreq(event: ActionEvent): guarda el contingut que hi ha en els elements d'afegir Freqüència en el moment de clicar el botó Guardar.
+ actionPerformed_buttonCancelAlfabet(event: ActionEvent): torna a la vista principal sense guardar l'alfabet que s'estava creant.
+ actionPerformed_buttonCancelFreq(event: ActionEvent): torna a la vista principal sense guardar la freqüència que s'estava creant.
+ actionPerformed_buttonCancelDades(event: ActionEvent): torna a la vista principal sense guardar les dades que s'estaven creant.
+ actionPerformed_buttonCancelTeclat(event: ActionEvent): torna a la vista principal i cancela la generació del teclat que s'estava creant.
+ actionPerformed_buttonGenerarTeclat(event: ActionEvent): mostra una finestra que permet seleccionar els elements que volem per crear un teclat.

- asignar_listenersComponents(): assigna a cada botó una acció.
- inicialitzarComponents(): crida les altres funcions inicialitzadores.
- inicialitzar_frameVista(): inicialitza el frame de la vista secundària.
- inicialitzar_panelAfegirAlfabet(): afegeix al panel d'afegir Alfabet els elements perquè l'usuari pugui entrar l'alfabet.
- inicialitzar_panelAfegirFrequencia(): afegeix al panel d'afegir Freqüència els elements perquè l'usuari pugui escollir un nom i escollir les dades d'on treure una freqüència de símbols.
- inicialitzar_panelAfegirTeclat(): afegeix al panel d'afegir Teclat els elements perquè l'usuari pugui escollir un nom i quines dades vol utilitzar per generar un teclat.
- inicialitzar_panelMostrarPuntuacio(): inicialitza el panell per mostrar puntuació.

- desactivaComponentsGeneracioTeclat(): desactiva tots els components del panel d'afegir teclat menys el botó cancelar.
- netejar_panelAfegirTeclat(): torna a activar els components necessaris del panel d'afegir teclat i posa les seleccions per defecte.

1.2.3. VistaTerciaria

La vista terciària és utilitzada a l'hora de crear un nou teclat. Aquesta permet a l'usuari visualitzar les freqüències, textos i llistes de paraules que podrà seleccionar.
Aquesta classe es compon de 5 aspectes principals.

1. Atributs de la Classe:
    - Es van utilitzar components de GUI com JButton per a representar els botons "Seleccionar" per a les pantalles de selecció d'alfabets, freqüències o textos/llestes de paraules.
    - DefaultListModel juntament amb JList s'utilitzen per a gestionar i mostrar aquests objectes.

2. Constructor de la VistaTerciaria:
    - La classe compta amb una constructora que s'encarrega de la inicialització.

3. Mètodes Públics:
    - Inclou mètodes com "seleccionarX()", on X pot ser dades/text/freqüència/alfabet, que a través del ControladorPresentacio, faciliten l'obtenció d'alfabets, freqüències, textos/llestes de paraules i teclats per a la seva presentació en seleccionar dades.

4. Listeners:
    - S'utilitzen per a reaccionar davant els esdeveniments de clic sobre botons.
    - En concret sobre els botons de seleccionar alfabet/freqüències/textos i llistes de paraules. En prémer aquests botons els listeners criden al ControladorPresentacio per a retornar aquests objectes i després retornar a la vista secundària.

5. Mètodes Privats:
    - Aquests inclouen els per defecte per a assignar listeners i inicialitzar components.
    - També inclouen aquells propis com inicialitzar_panelDades(), inicialitzar_panelAlfabet(), inicialitzar_panelTexts() i inicialitzar_panelFrequencies() que inicialitzen el panell que veurà l'usuari a l'hora de seleccionar dades per a un teclat.

Atributs:
```
private ControladorPresentacio ctrlPresentacio;

private JFrame frameVista = new JFrame();
private JPanel panelContinguts = new JPanel();
private JPanel panelActual = new JPanel();

private JPanel panelLlistaDades = new JPanel();
private JPanel panelOpcionsDades = new JPanel();
private DefaultListModel<String> modelDades = new DefaultListModel<String>();
```

```
private JList<String> listDades = new JList<String>(modelDades);
private JButton buttonSeleccionarDades = new JButton("Seleccionar");

private JPanel panelLlistaTexts = new JPanel();
private JPanel panelOpcionsTexts = new JPanel();
private DefaultListModel<String> modelTexts = new DefaultListModel<String>();
private JList<String> listTexts = new JList<String>(modelTexts);
private JButton buttonSeleccionarTexts = new JButton("Seleccionar");

private JPanel panelLlistaFreq = new JPanel();
private JPanel panelOpcionsFreq = new JPanel();
private DefaultListModel<String> modelFreq = new DefaultListModel<String>();
private JList<String> listFreq = new JList<String>(modelFreq);
private JButton buttonSeleccionarFreq = new JButton("Seleccionar");

private JPanel panelLlistaAlfabets = new JPanel();
private JPanel panelOpcionsAlfabets = new JPanel();
private DefaultListModel<String> modelAlfabets = new DefaultListModel<String>();
private JList<String> listAlfabets = new JList<String>(modelAlfabets);
private JButton buttonSeleccionarAlfabets = new JButton("Seleccionar");

JPanel panelLlistaAlfabets = new JPanel();
JPanel panelOpcionsAlfabets = new JPanel();
DefaultListModel<String> modelAlfabets = new DefaultListModel<String>();
JList<String> listAlfabets = new JList<String>(modelAlfabets);
JButton buttonSeleccionarAlfabets = new JButton("Seleccionar");
```

Mètodes:

+ VistaTerciaria(ctrlPresentacio: ControladorPresentacio): creadora donat un ControladorPresentacio
+ ferVisible(): fa visible la vista terciaria.
+ ferInvisible(): fa invisible la vista terciaria.
+ seleccionarDades(): selecciona el texts/llistes de paraules guardades per poder mostrarles per pantalla més endevant.
+ seleccionarFrequencia(): selecciona les freqüències guardades per poder mostrarles per pantalla més endevant.
+ seleccionarAlfabet(): selecciona els alfabets guardats per poder mostrarles per pantalla més endevant.

- mouse_listeners(): accions que s'activen al fer doble clic sobre un element d'una llista.
- asignar_listenersComponents(): listeners que s'activen al clicar el botó de seleccionar dades/freqüència/alfabet dintre de afegir teclat.
- inicialitzarComponents(): funció que crida a totes les altres funcions inicialitzadores.
- inicialitzar_frameVista(): inicialitza el frame.
- inicialitzar_panelDades(): inicialitza el panell on es mostren els texts/llistes de paraules que l'usuari pot seleccionar.
- inicialitzar_panelFrequencies(): inicialitza el panell on es mostren les freqüències que l'usuari pot seleccionar.

- inicialitzar_panelAlfabets(): inicialitza el panell on es mostren els alfabets que l'usuari pot seleccionar.

1.2.4. VistaDialeg

La vista diàleg ens permet mostrar pop-ups per pantalla per a preguntar a l'usuari si vol guardar o no, esborrar o no, avisar sobre possibles excepcions durant l'ús de l'aplicació i per a oferir a l'usuari opcions.

Mètodes:
+ vistaDialeg(): creadora
+ setDialeg(titol: String, text: String, botons: String[], tipus: String): ensenya una pestanya *pop-up* de tipus *tipus*, on es mostra un text i unes opcions donades per el paràmetre *botons* i retorna el índex del botó clicat.

2. Diagrama del controlador i de les classes de la capa de domini

2.1. Explicació del controlador de domini

2.1.1 ControladorDomini

Conté les funcions necessàries perquè un Driver o una aplicació que faci servir el domini tingui accés a les funcionalitats especificades als casos d'ús (i algunes més).

Mètodes

generaTeclat(): té tres variants, una que accepta els noms d'un alfabet i una freqüència de símbols, una que accepta el nom d'una freqüència de símbols i una que accepta el nom d'un text o una llista de paraules amb freqüències (dades). S'especifica el mètode de generació amb una String, on les opcions són "CercaLocal" i "BranchAndBound".
visualitzaX(): retorna una String que representa els diferents elements del programa: (Teclat, FrequenciaSimbols, Alfabet, Dades). Serveix per veure per consola.
esborraX(): es dona el nom d'un element (pot set Teclat, Alfabet, FrequenciaSimbols, DadesParaules) i s'esborra.
afegirX(): Ídem, però per afegir. Per DadesParaules, es dona el path d'un fitxer.
afegirXDirectament [Text, LlistaDeParaules](): Per afegir DadesParaules de forma directa (en comptes de donar un path, es dona el contingut en una String.
modificaX [Teclat, Alfabet, DadesParaules, FreqSimbols](): permet fer canvis un dels elements existents al sistema.
getNomsX[Teclats, Alfabets, Dades, Freqs](): obté els noms de tots els elements de tipus X que el sistema té actualment.
getInfoTeclat(nomTeclat): específicament per poder visualitzar teclats a la capa de presentació, però sense passar objectes de tipus Teclat. Aquesta funció retorna un ArrayList de Strings amb la informació necessària per fer-ho.

2.2. Explicació de les classes de domini

2.2.1. LlistaTeclats

Atributs
Un *hashmap* de Teclats indexats pel seu nom (String).

Mètodes
size(): retorna el nombre d'elements.
clear(): buida la Llista.
getTeclat(nom): es retorna el Teclat identificat per "nom".
existeixTeclat(nom): retorna si existeix un Teclat identificat per "nom".
setTeclat(Teclat, nom): permet substituir el Teclat identificar per "nom" pel Teclat passat per paràmetre.
afegeixTeclat(Teclat): afegeix el Teclat, identificat pel seu propi nom.
esborraTeclat(nom)
getNoms(): retorna els noms de tots els elements de la Llista.

2.2.2. Teclat

Atributs
El nom.
Una assignació de caràcters (on el caràcter ièsim està assignat a la posició i del Layout).

Mètodes
toString(): Formateja el Teclat de forma que es veu per text com estan col·locades les tecles i amb quins símbols cadascuna.
toStringProcessable(): Formateja el Teclat en un ArrayList de Strings amb les dades necessàries per veure el teclat a Presentació: assignació de caràcters a tecles, posicions i amplades.
intercanviaSimbols(c1, c2): Serveix per modificar el Teclat, intercanvia les posicions dels símbols c1 i c2.
avalua(Text): Retorna la distància total recorreguda per tal d'escriure el Text amb aquest Teclat.

2.2.3. Layout

Atributs
distancies: una taula/matriu que guarda distàncies entre tecles.
nTecles: el nombre de tecles "oficials" que té (les de l'alfabet).
files: el nombre de files.
lletresPerFila: un array que per cada fila ens diu quantes tecles té.
posX i posY: utilitzades per calcular distàncies i per donar les posicions de cada tecla (necessari a presentació).

Atributs estàtics privats, constants per tots els layouts:

AMPLE i ALT d'una tecla (actualment 1.0), TECLES_ESPECIALS (actualment 6) i ASSIGNACIO_ESPECIAL, que són les tecles de l'última fila: el menú especial, majúscules, l'espai, la coma, el punt i la tecla d'esborrar.
MIDA_NORMAL_FILA: val 10. És el nombre de tecles que té una fila sencera del Teclat.
AMPLADA_X [Menu (1.0), majus (1.5), espai (4.0), coma (1.0), punt (1.0), esborrar (1.5)]. L'amplada de cada tecla de la fila especial.
AMPLADES, un array amb les variables AMPLADA_X. Sumades totes, dona 10.0.
ASSIGNACIO_SECUNDARI, que és l'assignació de tecles del teclat secundari: els dígits, @,#,€,_,&,-,+,(,),/,*,",',:,;,!,?,´,`,%. Són 30 en total, per tal que facin 3 files de 10.
POS[X|Y]_SECUNDARI, arrays amb les coodenades de les tecles del layout secundari.
ASSIGNACIO_SECUNDARI_ESPECIAL, que és l'assignació de tecles a l'última fila del layout secundari: retornar al layout principal, l'espai, coma, punt i esborrar.
AMPLADES_SECUNDARI, que és un array com amplades, però per la fila especial del layout secundari. També sumen 10.

Mètodes
distanciaEntre(p1,p2): retorna la distància entre les posicions p1 i p2.

La resta són consultores d'atributs.

2.2.4. DadesParaules

És una classe polimorfista que no té atributs, de la qual pengen les classes que representen les llistes de paraules i els textos
Mètodes
size(): indica quants caràcters/paraules té l'objecte.
getFrequencia(índex): per una llista de paraules retorna la freqüència de la paraula en posició índex.
getParaules(índex): per una llista de paraules retorna la paraula en posició índex.
getSimbol(índex): per un text retorna el caràcter en posició índex.
getNom(): retorna l'identificador/nom de l'objecte.
toString(): retorna el contingut de l'objecte.
getAlfabet(): retorna l'alfabet que s'ha utilitzat per a l'objecte
copy(): fa una copia de l'element
setNom(name): canvia el nom de l'element a *name.*
setContingut(contingut): canvia el contingut a *contingut*.
numSimbol(): retorna el nombre de símbols en un text.

2.2.5. FreqParaules

Atributs
nom: string que conté l'identificador de l'element.
paraules: vector on, per cada posició, es guarda una paraula amb la seva freqüència.

2.2.6. Text

Atributs
nom: string que conté l'identificador de l'element.
text: string que conté el text.
paraules: vector on a cada posició conté una paraula del text.

2.2.7. LlistaDadesParaules

Atributs
texts: *hashmap* de DadesParaules indexades pel seu nom

Mètodes
size(): retorna el nombre d'elements.
exists(nom): retorna si l'element "nom" existeix dins la llista o no.
getNoms(): retorna un vector amb els noms de tots els elements de la llista.
getNomsTexts(): retorna un vector amb els noms de tots els textos de la llista.
getDadesParaules(nom): retorna l'element amb índex "nom".
setDadesParaules(nom,dades): permet substituir l'element DadesParaules identificat per "nom" per l'element DadesParaules passat per paràmetre.
addDadesParaules(dades): afegeix l'element dades identificat pel seu propi nom.
removeDadesParaules(nom): elimina l'element identificat per "nom".

2.2.8. Alfabet

Atributs
nom: String que representa el nom de l'alfabet.
alfabet: un HashSet de Characters que representa l'alfabet.

Mètodes
Aquesta classe també implementa les següents funcions:

- Alfabet(): creadora buida d'un alfabet.
- Alfabet(alfabet): creadora amb un set passat per paràmetre.
- Alfabet(nom, alfabet): creadora amb nom i set passats per paràmetre.
- copiar(): permet fer una còpia de l'alfabet.
- afegeixSimbol(símbol): afegeix a l'alfabet el símbol que li passem per paràmetre.
- esborraSimbol(símbol): esborra de l'alfabet el símbol que li passem per paràmetre.
- setAlfabet(alfabet): substitueix l'alfabet actual per aquell que li passem per paràmetre.
- setNom(nom): substitueix el nom actual per aquell que li passem per paràmetre.
- setSimbol(símbol, índex): substitueix el símbol en la posició índex pel símbol que li passem per paràmetre.
- mergeAlfabets(array alfabets): fusiona en un únic alfabet els alfabets passats per paràmetre (mitjançant un array d'alfabets).
- toString(): override de la funció toString() que transforma l'alfabet en un string.

- equals(alfabet): retorna true si l'alfabet actual conté els mateixos símbols que l'alfabet que passem per paràmetre.
- exists(símbol): retorna true si l'alfabet conté el símbol que passem per paràmetre.
- getSimbol(índex): retorna el símbol situat en la posició índex passat per paràmetre.
- getSize(): retorna la mida de l'alfabet.
- getNom(): retorna el nom de l'alfabet.

2.2.9. LlistaAlfabets

Atributs

alfabets: un HashMap de String, Alfabet que representa la llista d'alfabets amb nom (String) i valor (alfabet).

Mètodes

Aquesta classe implementa les següents funcions:

-afegeixAlfabet(alfabet): afegeix al mapa l'alfabet que passem per paràmetre.

-esborraAlfabet(nom): esborra del mapa l'alfabet amb el nom que passem per paràmetre.

-setAlfabet(nom, alfabet): substitueix l'alfabet amb el nom passat per paràmetre pel nou que passem per paràmetre.

-getSize(): retorna la grandària del mapa.

-exists(nom): retorna true si el mapa conté l'alfabet amb el nom que passem per paràmetre.

-getAlfabet(nom): retorna l'alfabet amb el nom que passem per paràmetre.

-getNoms(): retorna un array amb els noms de tots els alfabets en el mapa.

2.2.10. FrequenciesSimbols

Atributs

nom: el nom de la freqüència. Serveix per identificar-la.

frequencies: matriu per guardar la freqüència (nombre de vegades que surt un adjacent a l'altre) entre tot parell de símbols.

traduccio: estructura per passar de símbols a índexs de la matriu *frequencies*.

Mètodes

frequencia(c1, c2): retorna la freqüència entre els símbols c1 i c2.

copia(freq2): retorna un objecte FrequenciesSimbols equivalent a freq2 però sense repetir referències, creant un objecte nou.

toString(): retorna el contingut de la matriu *frequencies* transformat a String.

2.2.11. LlistaFreqSimbols

Atributs

frequencies: *HashMap* de freqüències indexades pel seu nom.

Mètodes

size(): retorna el nombre d'elements.

afegeixFrequencies(freq): afegeix *freq* a *frequencies*, identificada pel seu nom.

setFrequencies(nom, freq): substitueix la freqüència identificada per *nom* per *freq*.
eliminaFrequencies(nom): elimina la freqüència identificada per *nom*.
getNoms(): retorna els noms de totes les freqüències en una llista.
getFrequenciesSimbols(nom): retorna la freqüència identificada per *nom*.

2.2.12. AlgorismeGeneracio

La classe serveix per definir la interfície que ha d'implementar qualsevol algorisme que resolgui el problema de la pràctica. Concretament, han de tenir la funció *executa*, que donats un alfabet, una freqüència de símbols, i un layout retorna un array de caràcters de la mida de l'alfabet.

2.2.13. CercaLocal

És un dels dos mètodes possibles d'optimització. Rep a la constructora el nombre d'iteracions per Hill Climbing i només implementa la funció *executa*.

2.2.14. Classes base per Hill Climbing

La classe CercaHillClimbing fa servir estats programats per l'usuari de l'algorisme, i com es fa un estat diferent per cada problema, es tractaran com a Objects. Aquesta classe rep un estat inicial, una funció heurística i una funció generadora de successors particulars pel problema a resoldre. Es defineix una interfície de funció heurística i de funció generadora de successors. També es permet que els successors guardin una descripció de l'operador aplicat a cada pas, de forma que es pot saber què ha fet l'algorisme si fos necessari.
Com que la constructora fa la cerca, un cop es construeix l'objecte només hem de consultar l'estat final.

2.2.15. EstatHillClimbing

Atributs
freq: freqüència entre els símbols a col·locar en l'assignació.
layout: layout del teclat a generar
assignacio: array de caràcters que guarda la solució de l'algorisme

Mètodes
funcioHeuristica(): retorna la qualitat de l'estat (solució). Un valor menor representa una millor qualitat.
creaSolucioInicial(alfabet, seed): serveix per generar una assignació inicial. És aleatòria.
intercanvia2Caracters(i, j): funció per canviar la solució actual.
copia(estat): retorna un EstatHillClimbing nou equivalent a *estat*, amb la seva pròpia còpia d'*assignació*.
getNumCaracters(): retorna la mida d'*assignacio*.

2.2.16. QAP

Atributs
estat[]: array de ints que representa l'assignació actual de les tecles (índex) en les diferents ubicacions (valor).

sol_aux[]: array d'ints que representa la millor solució completa trobada fins al moment.
posOcupada[]: array de booleans que indica quines columnes estan assignades.
cota_max: double que representa la cota màxima que no podem superar.

Mètodes
executa(Alfabet alfabet, FreqSimbols fs, Layout layout): aquesta funció serveix per executar l'algorisme Branch&Bound.
Greedy(Alfabet alfabet, FreqSimbols fs, Layout layout): aquest algorisme serveix per calcular una cota_max inicial.

2.2.17. GilmoreLawler

Mètodes
Bound(HungarianAlgorithm HA, int[] estat, boolean[] posOcupada, Alfabet alfabet, Layout layout, FreqSimbols fs): aquesta funció ens serveix per calcular el bound utilitzat al Branch&Bound per podar solucions parcials.

2.2.18. HungarianAlgorithm

Mètodes
HungarianAlgorithmOPT(double[][] C, int size): aquesta funció la fem servir per calcular l'assignació òptima de la matriu de costos C.
PreProcess(double[][] C_aux, int size): aquesta funció la fem servir per restar el valor mínim de cada fila i columna de la matriu C_aux.
maxAssig(double[][] C_aux, int size): aquesta funció la fem des de la funció HungarianAlgorithmOPT servir per trobar l'assignació màxima de la matriu C_aux.
Assig(): aquesta funció la fem servir des de la funció maxAssig per determinar si una tecla es pot assignar a una ubicació.

3. Diagrama de les classes i controladors de la capa de persistencia

3.1. Explicació del controlador de persistència

La persistència està implementada amb una única classe gestora, ControladorPersistencia, que s'encarrega de carregar i desar totes les dades. Es defineixen quatre fitxers:

- teclats.dat
- alfabets.dat
- dades.dat
- frequencies.dat

On tindrem les respectives llistes. Estaran situats al directori EXE de l'estructura de directoris de l'entrega, dins un altre subdirectori "dades".

La granularitat és a nivell de llista: per accedir a un element, per exemple, de tipus Teclat, hem de carregar la LlistaTeclats, i de la mateixa forma es fa per desar.

Hi ha les següents funcions:

De càrrega: una per cada element
- carregaTeclats()
- carregaAlfabets()
- carregaDades()
- carregaFrequencies
- carregaObj(path)

De desat: una per cada element
- desaTeclats()
- desaAlfabets()
- desaDades()
- desaFrequencies()
- desaObj(path, objecte)

Definim funcions genèriques amb "Obj" per reaprofitar codi. Com es pretén fer servir l'interfície Serializable de Java, no cal implementar res més a les classes del domini o aquí.