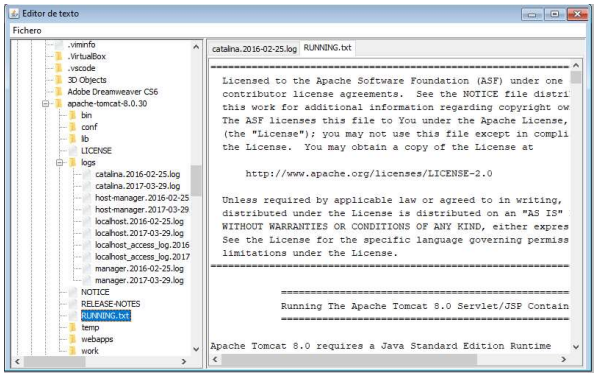
Práctica: Visor de ficheros de texto

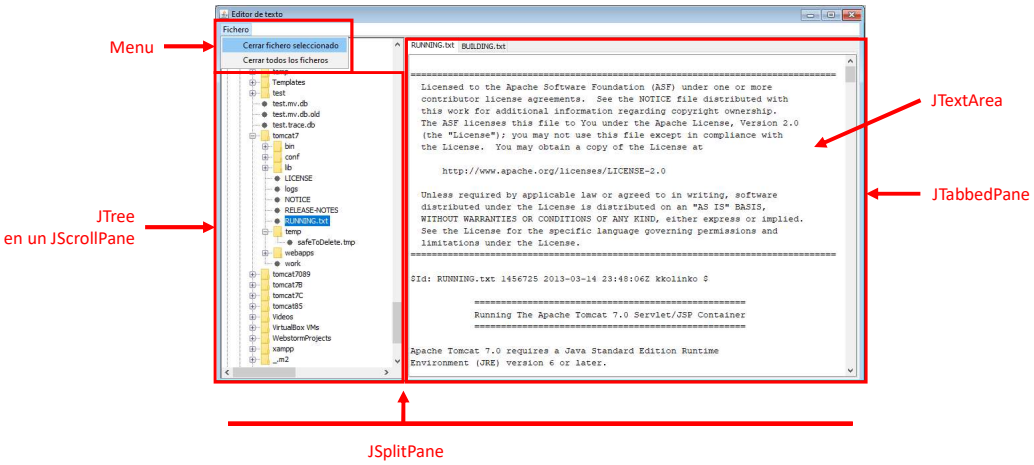
- Descripción y diseño
- Componentes
- Desarrollo

Descripción



- Visor de ficheros de texto multipestaña
- Carga la estructura de ficheros y carpetas en un JTree
  - Carga progresiva de carpetas anidadas al expandir los nodos
- Permite abrir un fichero haciendo doble click en él
- Menús para cerrar una o todas las pestañas

Diseño



Componentes y eventos

- JMenuBar / JMenu / JMenuItem
- JTree
- JTextArea
- JSplitPane
- JScrollPane
- JTabbedPane
- ActionEvent / ActionListener
- MouseEvent / MouseListener (doble click)

Desarrollo

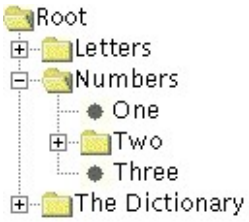
- Creación de la clase derivada de JFrame
- Diseñar la interfaz:
  - Agregar menu (JMenuBar)
  - Agregar panel de contenido (JSplitPane)

Desarrollo

- Inicializar el árbol de directorio en "C:/"
  - Crear nodo raíz con objeto File → C:/
  - Crear objeto JTree con el nodo raíz
  - Cargar subcarpetas y ficheros como nodos hijos (1 nivel)
- Implementar expansión de carpetas:
  - Listener MouseListener (doble click)

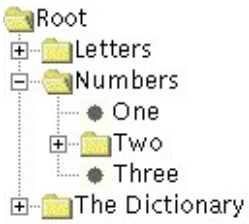
Arboles (JTree)

- Componentes:
  - Arbol → **JTree**
  - Nodo → **TreeNode** (*DefaultMutableTreeNode*)
- Se construye :
  - A partir de un **nodo raíz**
  - O con un HashTable, array de objetos o Vector
- Nodos hijos:
  - Tipo "Rama": pueden tener a su vez nodos hijos
  - Tipo "Hoja"(*leaf*): no tienen hijos, son nodos finales
- Los nodos se cargan recursivamente con ***add()***
- Tipos de nodos según contenido asociado:
  - Nodo simple: texto
  - Nodo complejo: objeto de cualquier tipo. Como texto en el árbol se aplica "*toString()*" del objeto.



Arboles (JTree)

- Otras capacidades:
  - Selección simple o múltiple
  - Arrastrar y soltar
  - L&F define iconos por defecto, pero se pueden personalizar
  - El texto de cada nodo puede ser editado por el usuario
- Eventos:
  - Cambio de nodo seleccionado
  - Expandir o comprimir cualquier nodo
  - Añadir, modificar o quitar nodos



JTree

Permite mostrar y navegar a través de datos relacionados de forma jerarquica, como nodos desplegables en árbol.

```
// Crear un arbol a partir de un nodo raiz
DefaultMutableTreeNode nodoRaiz = new DefaultMutableTreeNode("Raiz");
JTree arbol1 = new JTree(nodoRaiz);

// Añadir nodo de texto al nodo raiz
nodoRaiz.add(new DefaultMutableTreeNode("Hijo"));

// Añadir un nodo de cualquier tipo de objeto
nodoRaiz.add(new DefaultMutableTreeNode(new Partido())); // Usa toString()
   para mostrar el texto

// Crear un arbol a partir de un array de objetos:
String[] lista = new String[]{"Uno", "Dos", "Tres");
JTree arbol2 = new JTree(lista); // No muestra el nodo raiz

// Obtener el nodo seleccionado
DefaultMutableTreeNode nodoSeleccionado = (DefaultMutableTreeNode)
   arbol2.getLastSelectedPathComponent();

// Leer el objeto asociado al nodo seleccionado
(String) valorNodo = (String) nodoSeleccionado.getUserObject();

// Leer hijo en la posicion N del nodo
DefaultMutableTreeNode hijo = nodoseleccionado.getChildAt(N-1);

// Borrar hijo N del nodo
nodoseleccionado.remove(N-1);
```

- Métodos
  - **setRootVisible()**: Visibilidad del nodo raíz
  - **setCellRenderer()**: Personalizar aspecto de cada nodo
  - **setEditable()**: Texto del nodo editable por el usuario
  - **setDragEnabled()**: Permitir arrastrar nodos
  - **set/getSelectionPath()**: Ruta seleccionada (*TreePath*)
  - **getLastSelectedPathComponent()**: Nodo seleccionado
  - **expandPath() / collapsePath()**: Expandir o contraer rutas.
  - **scrollPathToVisible()**: Expande una ruta para que sea visible.
  - **getNextMatch()**: Buscar un nodo con un prefijo
- Eventos / Listeners
  - TreeSelectionListener:
    - valueChanged()
  - TreeExpansionListener:
    - treeCollapsed(), treeExpanded()
  - TreeWillExpandListener():
    - treeWilCollapse(), treeWillExpand()

Desarrollo

- Abrir un fichero de texto en una pestaña del TabbedPane:
  - MouseListener para el doble click sobre el árbol
  - Funcion abrirFichero:
    - Solo debe abrir los ficheros de texto permitidos
    - Crear TextArea
    - Cargar fichero en TextArea
    - Meter el TextArea en un ScrollPane
    - Añadir una pestaña con el ScrollPane y el nombre del fichero

JTextArea

Editor de texto plano multilinea. No soporta texto con diferentes formatos o estilos. Para mostrar textos largos requiere estar dentro de un JScrollPane.

```
// Leer texto de un fichero
JTextArea editor = new JTextArea();
editor.setWrapStyleWord(true);

File fichero = new File("C:/miFichero.txt");

BufferedReader br;
try {
   br = new BufferedReader(new FileReader(fichero));
   String linea;
   while ((linea = br.readLine()) != null) {
            editor.append(linea + "\n");
   }
} catch (IOException e) {
            e.printStackTrace();
}
```

- Métodos
  - **set / getText()**: Establece o lee el texto
  - append(): Añade texto al final
  - insert(): Inserta texto en una posición
  - replaceRange(): Reemplaza texto entre 2 posiciones
  - setEditable(): Texto editable
  - setWrapStyleWord(): Modo de salto de línea. True= salta en los espacios entre palabras.
  - selectAll(): Selecciona todos los caracteres
- Eventos / Listeners
  - No dispone de eventos o listener específicos

Desarrollo

- Implementar opciones de menu para cerrar pestañas:
  - Asignar "actionCommand" a cada elemento de menu
  - Crear ActionListener y gestionar los eventos

Desarrollo

- Mostrar solo nombre de fichero en lugar de ruta completa:
  - Crear una subclase de File
  - **Sobreescribir el método *toString()*** para que devuelva el "name"
  - Modificar "cargarFicherosCarpeta()" para que los objetos que se añadan al árbol sean de tipo "FileArbol" en lugar de "File"