**Conocimiento: AWT- Barra de Menu**

En las ventanas de interfaz gráfica, no solo se agregan componentes sino que también se pueden acceder a menues

Un menú en una aplicación no es más que un MenuBar en el que hay varios menús.

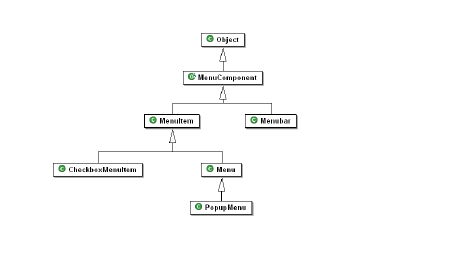
Si se piensa en un programa cualquiera con las opciones de menú Archivo, Editar y Ayuda, estas tres opciones en Java son unos objetos de la clase Menú y se tienen que añadir a un objeto de la clase MenuBar que se une a la ventana. Cada menú tiene varias opciones, por ejemplo, el menú Archivo tendrá las opciones: Abrir, Cerrar, Guarda y Salir. Éstos en Java son unos objetos de la clase MenuItem (o también Menú si incluyen otros submenús).



Por lo tanto, si una aplicación necesita añadir un menú se debe realizar las siguientes actividades:

1. Crear los objetos MenuItem
2. Crear los objetos menú y agregarle los MenuItem
3. Crear una MenuBar y agregarle los Menús
4. Agregar el MenuBar al contenedor que acepte un MenuBar, por ejemplo: Frame.

Jerarquía de Clases:



**Clase MenuComponent**

Esta clase es abstracta y es la super clase de la jerarquía de clases que administran los Menús en Java. Como es abstracta no se instancia pero constituye la base para todos los componentes de Menu.

* Constructor:
  + - **MenuComponent ():** este constructor crea un nuevo MenuComponent.
* Utilidades: esta clase hereda los métodos de la clase Object, además agrega nuevas utilidades tales como:
  + - **Font getFont ():** este método retorna la fuente de las letras del componente.
    - **String getName ():** este método retorna un String que contiene el nombre del componente.
    - **MenuContainer getParent ():** este método retorna el contendor padre donde se ubica el MenuComponent.
    - **void setFont (Font f):** este método permite cambiar el tipo de letra que usará el MenuComponent.
    - **void setName (String nombre):** este método permite cambiar el nombre del MenuComponent.

**Clase MenuItem**

Esta clase hereda de MenuComponent y se emplea para instanciar los objetos que constituirán los elementos seleccionables del menú.

* Constructor:
  + - **MenuItem ():** este constructor crea un nuevo MenuItem con una etiqueta vacía y sin tecla de acceso rápido.
    - **MenuItem (String etiqueta):** este constructor crea un nuevo MenuItem con una etiqueta determinada por el parámetro de entrada y sin tecla de acceso rápido.
    - **MenuItem (String etiqueta, MenuShortcut s):** este constructor crea un nuevo MenuItem con una etiqueta determinada por el parámetro de entrada y con una tecla de acceso rápido determinada por MenuShortcut.
* Utilidades: esta clase utiliza todos los métodos de MenuComponent, además agrega otras funcionalidades tales como:
  + - **void addActionListener (ActionListener listener):** este método agrega un escuchante de acción al presionar el MenuItem.
    - **String getLabel ():** este método retorna un String conteniendo la etiqueta del MenuItem.
    - **MenuShortcut getShortcut ():** este método retorna la tecla que accede rápidamente a este MenuItem.
    - **boolean isEnabled ():** este método verifica si el MenuItem está habilitado o deshabilitado.
    - **void setEnabled (boolean b):** este método permite habilitar o deshabilitar un MenuItem.
    - **void setLabel (String label):** este método permite cambiar la etiqueta del MenuItem.
    - **Void setShortcut (MenuShortcut s):** este método permite cambiar la tecla que accede rápidamente a este MenuItem.

**Clase Menu**

Esta clase hereda de l MenuItem y es la que se utiliza para construir los menús que se manejan habitualmente, conocidos como menús de persiana (o pull-down).

Esta clase está implementada como una subclase de MenuItem para se puedan crear submenús fácilmente añadiendo un menú a otro.

* Constructor:
  + - **Menu ():** este constructor crea un nuevo menu con una etiqueta vacía.

* + - **Menu (String etiqueta):** este constructor crea un nuevo menu con una etiqueta determinada por el parámetro de entrada.
* Utilidades: esta clase hereda de MenuItem y utiliza todos sus métodos, además añade otras funcionalidades, tales como:
  + - **void addSeparator ():** este método permite agregar una línea de separación dentro de los ítems des menu.
    - **MenuItem getItem (int indice):** este método permite recuperar el MenuItem de la posición apuntada por indice.
    - **int getItemCount ():** este método permite obtener el número de elementos en el Menu.
    - **void insert (MenuItem menuitem, int indice**): este método permite insertar un MenuItem de la posición indicada por indice.
    - **void insert (String label, int indice):** este método permite insertar una etiqueta en la posición indicada por indice dentro del Menu.
    - **void insertSeparator (int indice):** este método permite insertar un separador en la posición indicada por indice, el elemento en dicha posición se desplaza un lugar más abajo dentro del menu.
    - **void remove (int indice):** este método permite remover o eliminar el elemento en la posición indicada por indice.
    - **void remove(MenuComponent item):** este método permite remover o eliminar el MenuItem indicado en el parámetro de entrada, desde el Menu

**Clase CheckboxMenuItem**

Esta clase hereda de MenuItem y representa objetos que son opciones de menu y que tienen una casilla de verificación.

* Constructor:
  + - **CheckboxMenuItem():** este constructor crea CheckboxMenuItem con una etiqueta vacía

* + - **CheckboxMenuItem (String etiqueta):** este constructor crea CheckboxMenuItem con una etiqueta especificada en el parámetro de entrada.

* + - **CheckboxMenuItem String etiqueta, boolean estado):** este constructor crea CheckboxMenuItem con una etiqueta especificada en el parámetro de entrada y activado o desactivado según lo que indique la variable estado.
* Utilidades: esta clase puede utilizar todos los métodos heredados, además agrega otras utilidades, tales como:
  + - **void addItemListener (ItemListener listener):** este método agrega un objeto eschuchante de acción en el item.
    - **boolean getState ():** este método verifica el estado del CheckboxMenuItem, es decir si está activado o no.
    - **void setState (boolean b):** este método permite cambiar el estado de activado del CheckboxMenuItem.

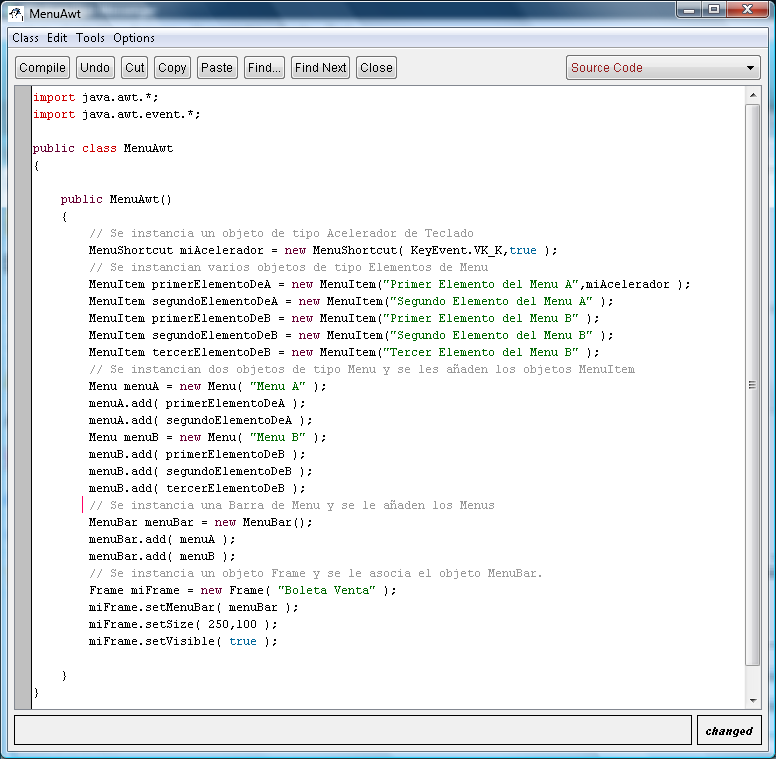
**Clase MenuBar**

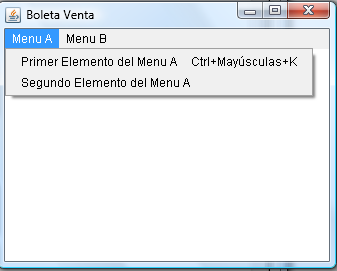
Esta clase hereda de MenuComponent y constituye la barra que recibe todos los menues que se agregarán al contenedor.

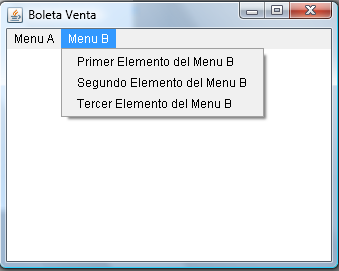
Las barras de menú no pueden utilizarse con la clase Panel.

* Constructor:
  + - **MenuBar ():** este constructor crea una nueva barra de menu vacía.
* Utilidades: utiliza todos los métodos de la clase MenuComponent, además agrega otras utilidades, tales como:
  + - **Menu add (Menu m):** este método permite agregar un menu a la barra.
    - **Menu getHelpMenu ():** este método permite obtener el menu de ayuda de esta barra de menu.
    - **Menu getMenu (int indice):** este método permite obtener el menu de la posición indice del parámetro de entrada.
    - **int getMenuCount ():** este método permite obtener la cantidad de menues en la barra de menu**.**
    - **void remove (int indice):** este método permite eliminar el menu número indice desde la barra de menu.
    - **void setHelpMenu (Menu m):** este método permite cambiar el menu de ayuda de la barra de menu.

Ejemplo de creación de un menu utilizando las clases antes vistas:

 Al ejecutar el código anterior el resultado es:





Aquí se muestran los dos menues al ser activados

**Clase PopupMenu**

Esta clase hereda de Menu y representa un componente que nos permite activar esos pequeños menues contextuales que aparecen al hacer un click con el botón derecho en alguna parte de una forma o ventana, o en alguno de los componentes que tiene un Frame (Ej.: panel, label, textfields, buttons, etc.).

