1. Menús e xanelas

1.1 Introdución

Na actividade que nos ocupa aprenderanse os seguintes conceptos e manexo de destrezas:

- Crear e xestionar nas nosas aplicacións gráficas de usuario os diferentes tipos de menús proporcionados pola linguaxe de programación java.
- Implementar aplicacións multixanela empregando diversas técnicas para a xestionar o entendemento entre as diferentes xanelas que poden compoñer as nosas aplicacións.
- Creación e emprego dos diálogos predefinidos proporcionados pola linguaxe co fin de axilizar a realización de tarefas habituais.

1.2 Actividade

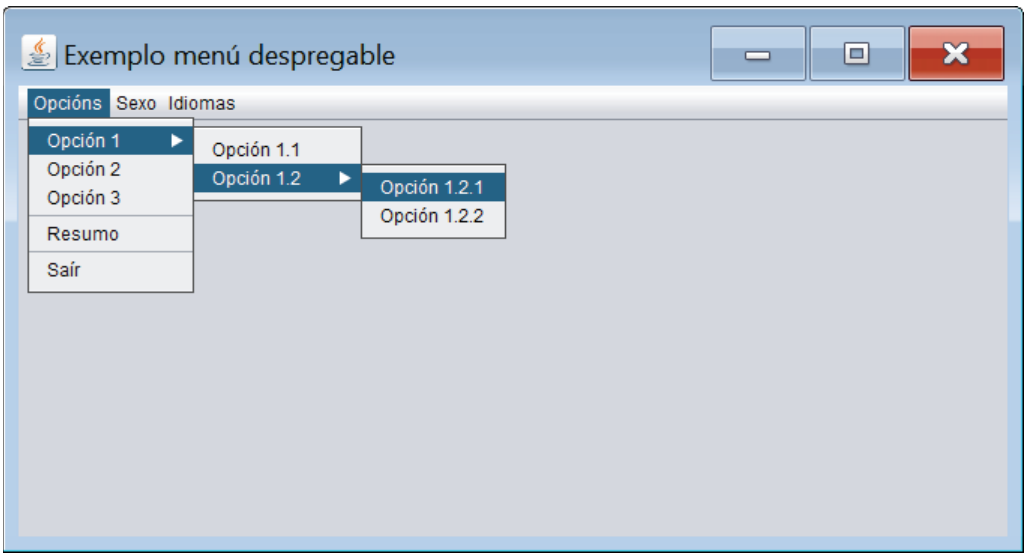
1.2.1 Menús

Un menú é unha representación gráfica ordenada dunha serie de opcións que nos proporciona unha aplicación. En función da opción seleccionada realizarase algunha acción asociada á selección. A linguaxe java proporciónanos dous tipos de menús, menús despregables e menús contextuais.

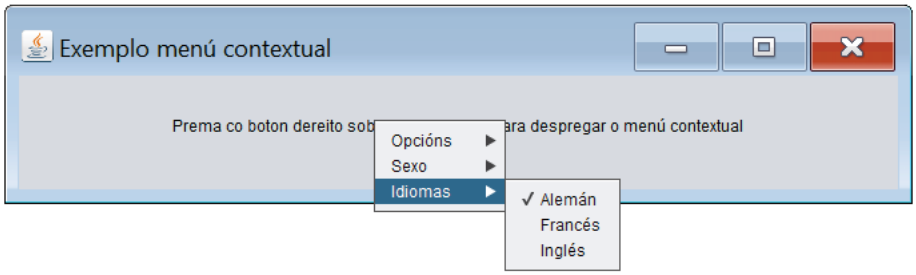
Os menús empregados habitualmente son de tipo menú despregable. Constan dunha serie de opcións organizadas nunha estrutura de árbore que colga dunha barra de menú. Á vez, a barra de menú colga dunha xanela (tipicamente do borde superior da xanela). Inicialmente só será visible a barra de menú, a cal conterá as opcións principais do menú. Ao premer sobre algunha destas opcións principais despregaranse os distintos submenús que colgan da barra de menú.

Pola outra banda, os menús contextuais inicialmente non son visibles. Constan dunha serie de opcións organizadas nunha estrutura de árbore que se visualizará unicamente ao realizar algunha acción (tipicamente clic dereito) sobre o compoñente que ten asociado o menú contextual.

A continuación amósase o aspecto visual dun menú despregable:



Nas seguinte imaxe pódese visualizar o aspecto dun menú contextual:



1.2.1.1 Menús despregables

Como acabamos de explicar un menú despregable consta basicamente dunha barra de menú da que colgan as opcións principais e de cada opción á vez poden colgar outros submenús (esta estrutura poderíase repetir recursivamente aínda que por motivos de claridade recoméndase non descender demasiados niveis na creación de submenús). A organización das distintas opcións que compoñen o menú ten unha estrutura de árbore no cal as ramas son accesos a submenús mentres que as follas son as opcións finais.

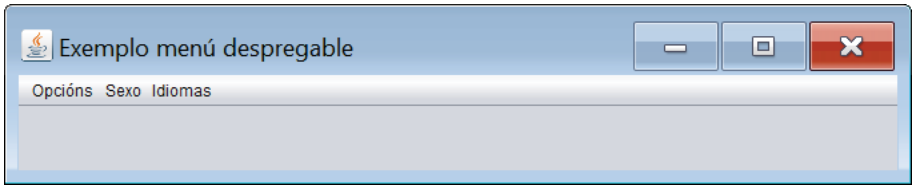
A creación de menús despregables baséase no emprego de obxectos das seguintes clases:

- javax.swing.JMenuBar: esta clase representa unha barra de menú
- javax.swing.JMenu: esta clase representa un menú. Dun menú pódense colgar máis JMenu (para a creación de submenús) o pódense colgar obxectos das clases JMenuItem, JRadioButtonMenuItem, JCheckBoxMenuItem e JSeparator. Os obxectos JMenu compoñen as ramas do árbore do menú, mentres que os obxetos das clases que imos comentar de seguido son as súas follas.
- javax.swing.JMenuItem: esta clase representa unha folla da árbore de opcións de menú. É unha opción á que se lle asocia a realización dunha acción ao ser seleccionada.
- javax.swing.JRadioButtonMenuItem: esta clase representa unha folla da árbore de opcións do menú. Compórtase como un botón de opción podendo estar ou non seleccionada entre un grupo de JRadioButtonMenuItems. É unha opción á que pódeselle asociar a realización dunha acción ao ser seleccionada ou deseleccionada, aínda que normalmente emprégase para indicar un valor de varios posibles, de xeito que a nosa aplicación funcione dun modo determinado en función desa selección.

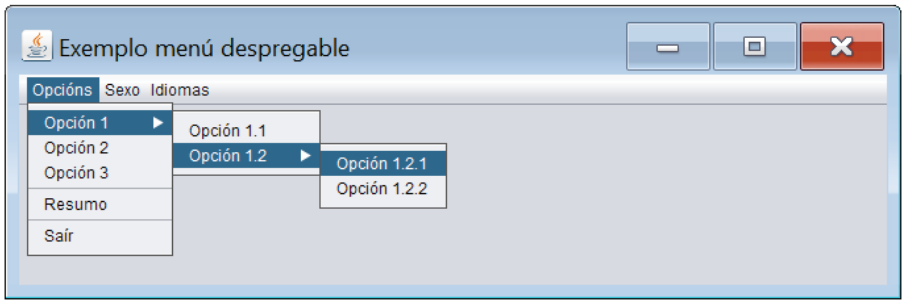
- javax.swing.JChekBoxMenuItem: esta clase representa unha folla da árbore de opcións do menú. Compórtase como unha casilla de verificación podendo estar ou non seleccionada. É unha opción á que pódeselle asociar a realización dunha acción ao ser seleccionada ou deseleccionada, aínda que normalmente emprégase para indicar se un valor está activo ou non, de xeito que a nosa aplicación funcione dun modo determinado en función desa situación.
- Javax.swing.JSeparator: esta clase representa unha liña e emprégase para separar visualmente dous opcións de menú entre si.

Configuración dun menú despregable empregando NetBeans

Co fin de explicar como crear un menú despregable empregando NetBeans imos desenvolver a seguinte aplicación:



Da opción principal Opcións colgarán os seguintes elementos:



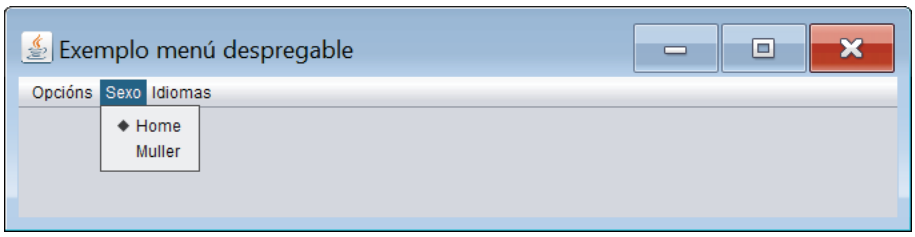
As opcións cunha frecha indican que son contedores de submenús (son obxectos de clase JMenu) de xeito que ao situarnos sobre elas co rato, ábrese o submenú asociado. Son ramas da árbore de opcións do menú.

O resto de opcións son follas da árbore de opcións do menú (son obxectos da clase JMenuItem):

- As opcións Opción 1.1, Opción 1.2.1, Opción 1.2.2, Opción 2 e Opción 3 ao ser premidas amosarán unha mensaxe indicando cal foi a opción seleccionada.
- A opción Resumo amosará por pantalla o sexo seleccionado no menú sexo e os idiomas seleccionados no menú Idiomas.
- A opción Saír finalizará a execución da aplicación.

Adicionalmente hai dous obxectos de clase JSeparator que empregaremos para separar entre si algunhas das opcións de menú.

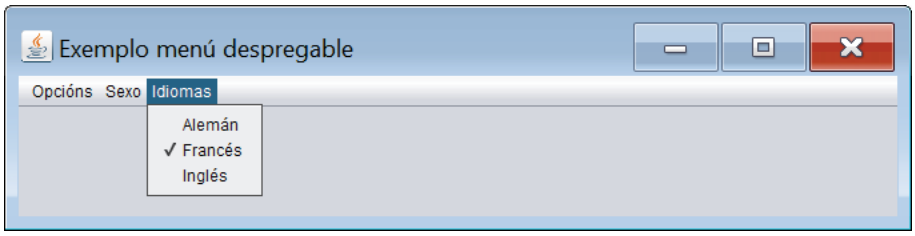
Da opción principal Sexo colgarán os seguintes elementos:



As dous opcións presentes neste menú son follas da árbore de opcións de menú (son obxectos da clase JRadioButtonMenuItem). Só unha delas pode estar seleccionada simultaneamente.

Cando elixamos a opción Home ademais de seleccionar esa opción, amosaremos unha mensaxe por pantalla indicando a selección realizada. Para a opción Muller faremos o mesmo.

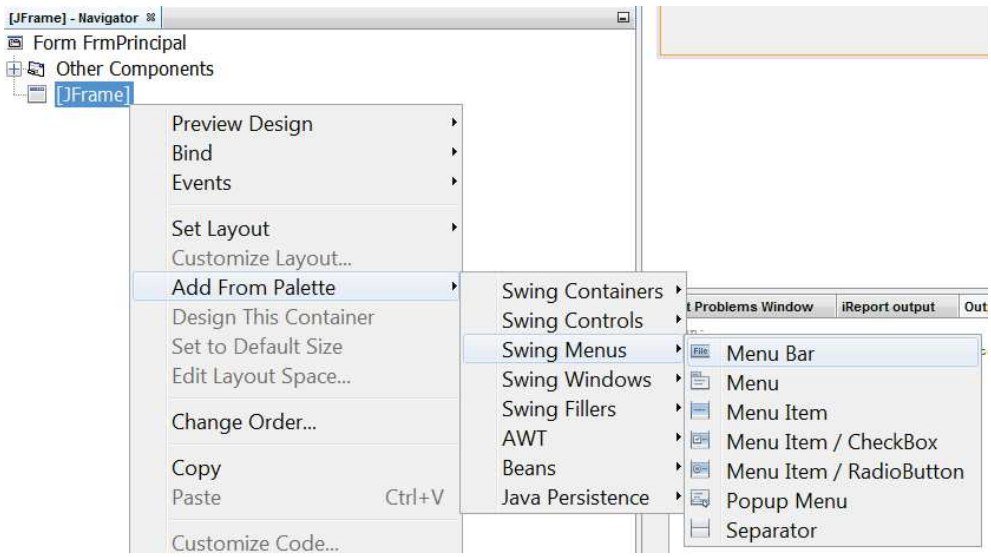
Por último da opción principal Idiomas colgarán os seguintes elementos:



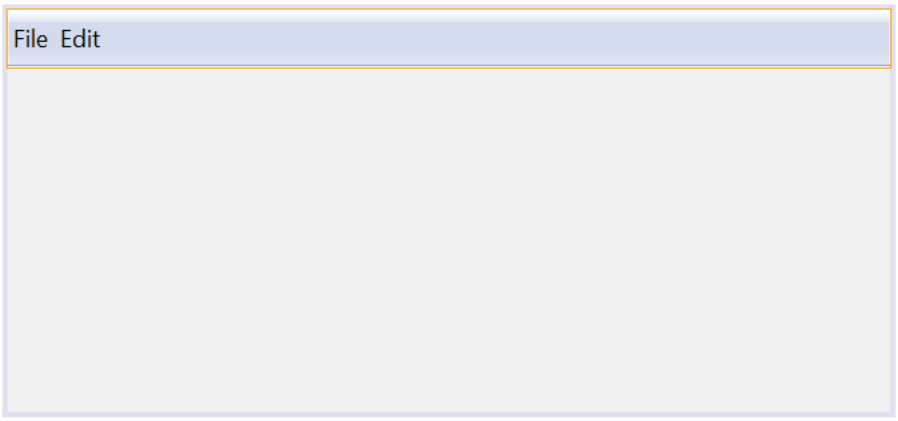
As tres opcións presentes neste menú son follas da árbore de opcións do menú (son obxectos da clase JCheckBoxMenuItem). Cada unha delas pode estar marcada ou desmarcada. Cando marquemos ou desmarquemos calquera delas amosaremos unha mensaxe por pantalla indicando a acción realizada.

Existen diferentes xeitos para xerar un menú empregando NetBeans. Aquí vaise explicar como facelo empregando a xanela de obxectos do formulario (xanela Navigator).

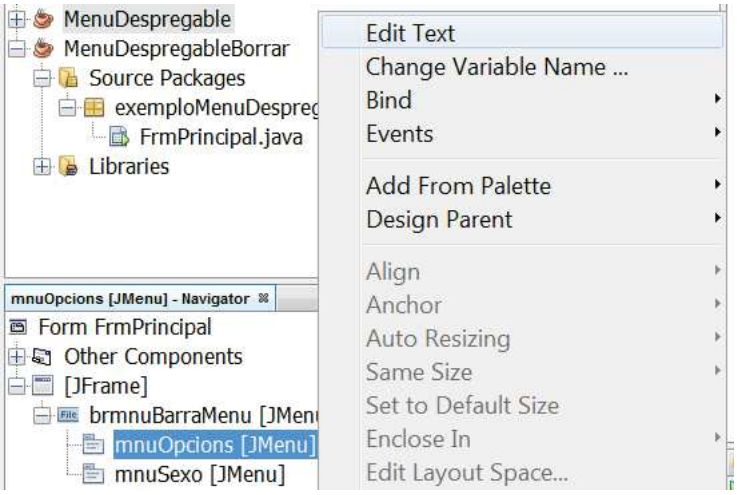
Partindo dun JFrame xa creado, imos engadirlle unha barra de menú. Para elo prememos co botón dereito do rato sobre o JFrame ao que lle queremos engadir a barra de menú e seleccionamos Add from Palette / Swing Menus / Menu Bar:



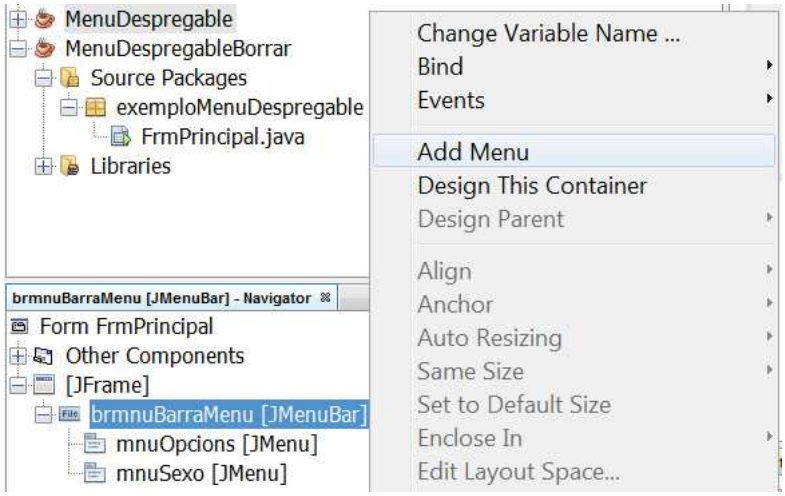
Ao realizar esta acción o entorno de programación engadirá ao formulario unha barra de menú con dúas opcións predefinidas (de clase JMenu):



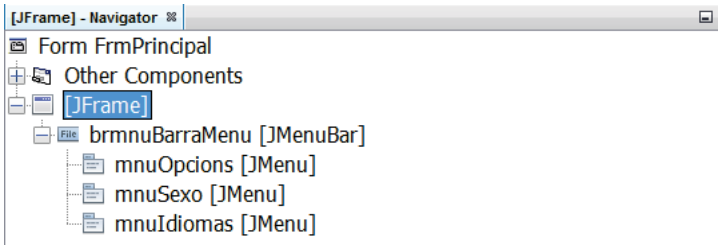
A estas dúas opcións de menú despois de darlles un nome interno máis descritivo, deberémoslle cambiar o texto. Imos ver como cambiar o texto que amosa o primeiro JMenu. Prememos co botón dereito do rato sobre o JMenu ao que queremos modificar o seu texto e seleccionamos a opción Edit Text. Por último modificamos o seu texto para que conteña o texto que desexamos que teña:



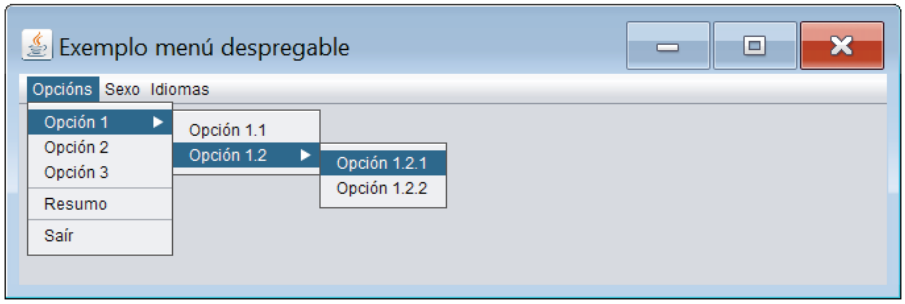
Agora mesmo temos dous opcións de menú pero precisamos de tres, polo tanto imos ver como engadir unha terceira. Como a opción de menú debe colgar da barra de menú, prememos co botón dereito sobre a barra de menú e seleccionamos a opción Add Menu. Unha vez engadido o terceiro menú modificaremos o seu nome interno e o seu texto.



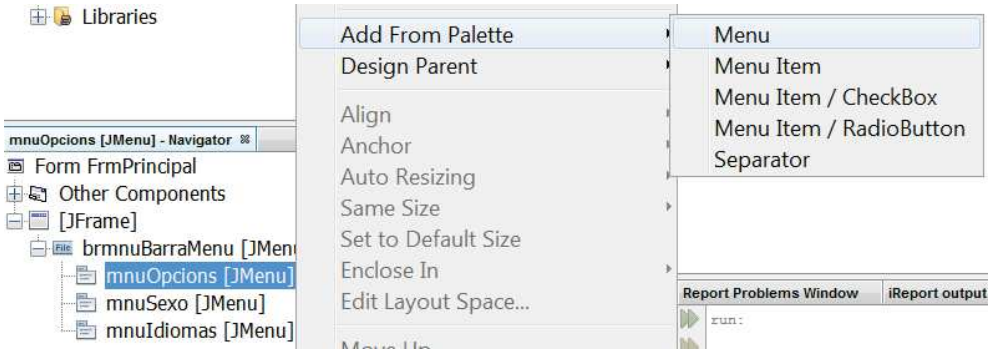
Deste xeito temos construídas tódalas opcións que colgan da barra de menú:



Agora ímonos centrar na rama que colga da opción Opcións.



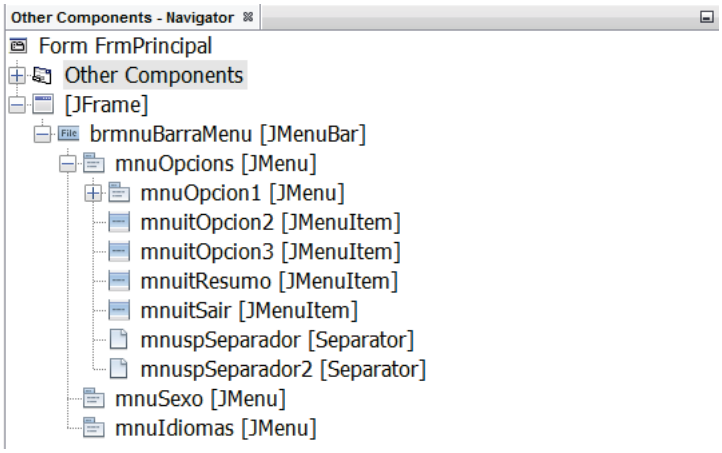
Opción 1 será un obxecto de clase JMenu (xa que vaise empregar para despregar un submenú) e colgará do obxecto mnuOpcions. Para crealo prememos co botón dereito do rato sobre o contedor da opción que imos crear e seleccionamos Add From Palette / Menu para indicar que queremos crear un JMenu.



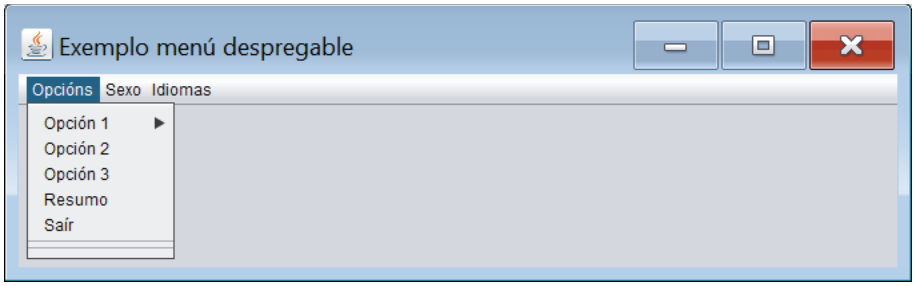
Non obstante, Opción 2 será un obxecto de clase JMenuItem (xa que vai ser unha folla da árbore do menú). Ademais tamén colgará do obxecto mnuOpcions. Para crealo prememos co botón dereito do rato sobre o contedor da opción que imos crear e seleccionamos Add From Palette / Menu Item para indicar que queremos crear un JMenuItem. Repetiremos este proceso para as opcións de menú Opción 3, Resumo e Saír.

Ademais temos que crear dous separadores de opcións de menú (obxectos da clase JSeparator). Para crear un separador prememos co botón dereito sobre o contedor do separador que imos crear e seleccionamos Add From Palette / Separator para indicar que queremos crear un JSeparator.

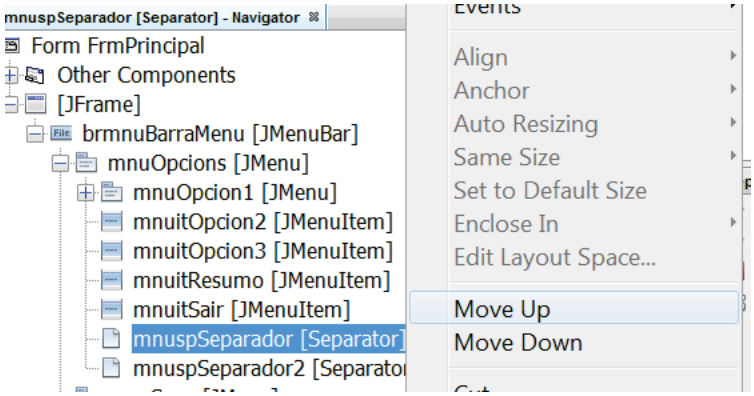
Unha vez creados estes elementos este é o resultado:



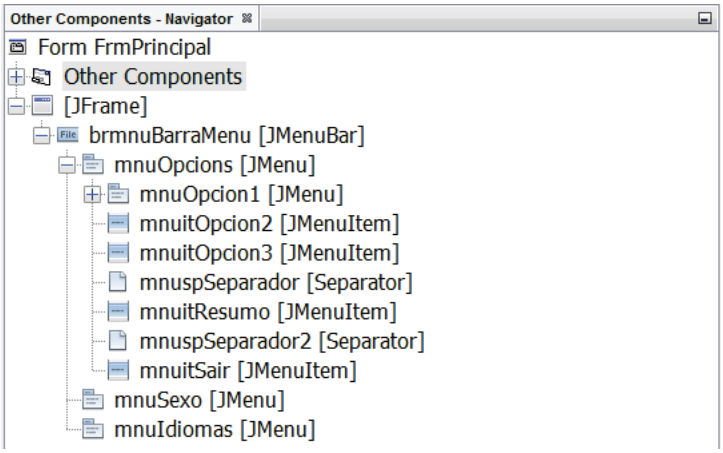
Se executamos a aplicación e prememos sobre o menú opcións observaremos o seguinte:



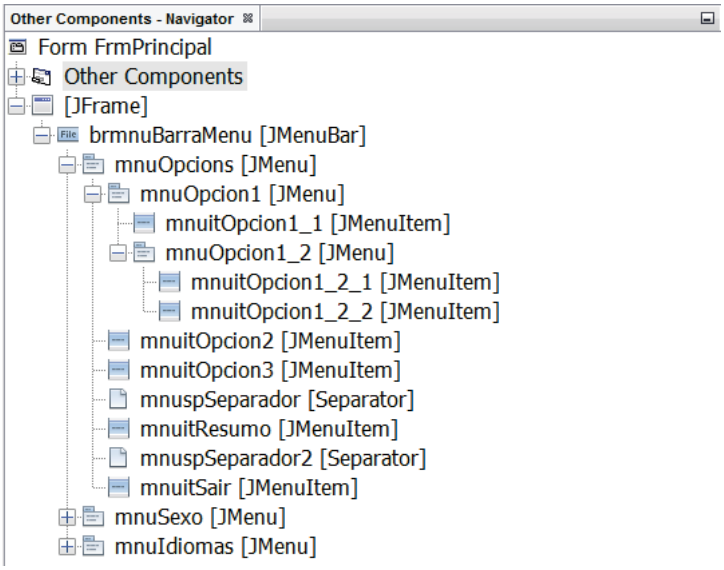
Os separadores aparecen despois das opcións. Isto é debido a que a orden dos elementos no menú é a mesma que a orde que ocupan na árbore de obxectos do proxecto. Neste caso vemos que na árbore do proxecto mnuspSeparador e mnuspSeparador2 aparecen ao final da súa rama e polo tanto ao executar a aplicación tamén aparecen ao final do menú. Para solucionalo deberemos colocar a mnuspSeparador e a mnuspSeparador2 no lugar que queremos que ocupen na árbore. A continuación imos ver como facer isto para mnuspSeparador. Prememos co botón dereito do rato sobre o elemento que queremos desprazar e prememos en Move Up. Con isto subiremos a mnuspSeparador unha posición na árbore de obxectos, pasando a estar entre mnuitResumo e mnuitSair. Repetiremos este proceso tantas veces como sexa necesario co fin de colocar cada elemento do menú no seu lugar. Se en lugar de subir un elemento quixéramolo baixar teremos que seleccionar a opción Move Down:



Finalmente esta é a configuración actual do noso menú:



Agora deberíamos crear os elementos do submenú que é despregado ao pasar sobre mnuOpcion1. Este submenú contén un JMenuItem (Opción 1.1) e un JMenu (Opción 1.2). Para crear o JMenuItem repetimos o proceso visto anteriormente fixándonos unicamente en que faremos clic co botón dereito do rato sobre o elemento que vai ser o seu contedor, é dicir, mnuOpcion1. O mesmo faremos para o JMenu. Unha vez creado este submenú crearemos o submenú que colga de Opción 1.2. Finalmente este será o resultado do menú Opcións:

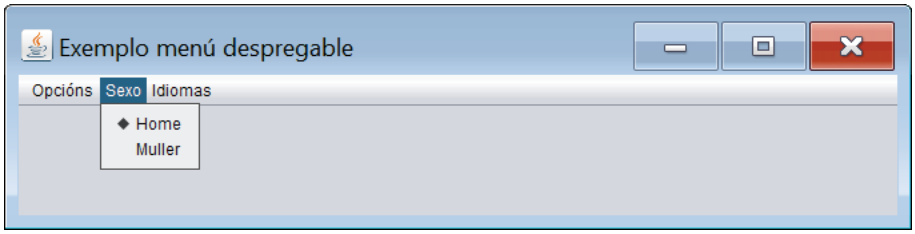


Unha vez resolta a parte visual imos implementar a súa funcionalidade. Os elementos da árbore que realizarán algunha acción son os elementos de clase JMenuItem. Imos ver como implementar a acción para a opción de menú mnuitOpcion1_1. Para elo temos que implementar o seu evento ActionPerformed. Para isto facemos dobre clic sobre o elemento mnuitOpcion1_1 na árbore de obxectos do proxecto e escribimos o seguinte código:

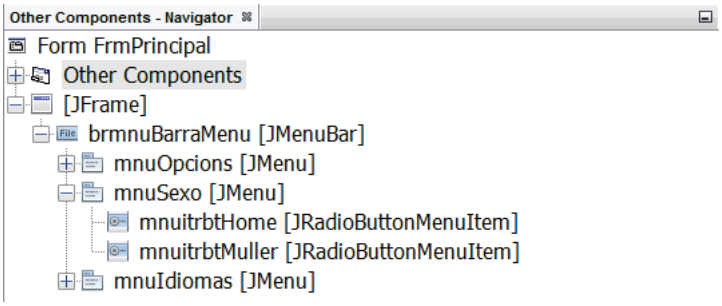
```
private void mnuitOpcion1_1ActionPerformed(java.awt.event.ActionEvent evt) {
    // TODO add your handling code here:
    JOptionPane.showMessageDialog(this, "Opción 1.1 seleccionada");
}
```

Como se pode observar no bloque de código anterior, no caso de que fagamos clic sobre a opción de menú mnuitOpcion1 amosarase por pantalla a mensaxe Opción 1.1 seleccionada.

De seguido ímonos centrar na rama que colga da opción Sexo.



Neste caso imos construír un menú cuxos compoñentes van ser JRadioButtonMenuItem. Procedemos do mesmo xeito que fixemos para o menú anterior, pero coa precaución de engadir elementos de clase JRadioButtonMenuItem. Unha vez creados esta será a configuración do menú:

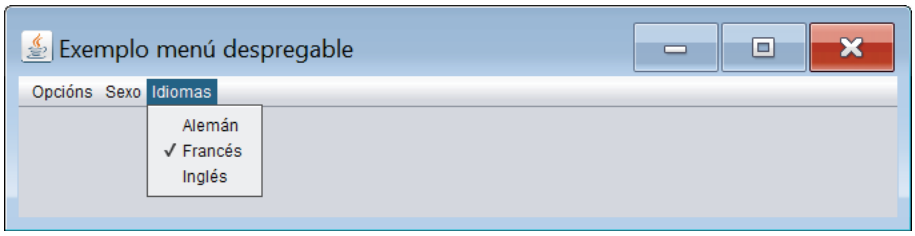


Os obxectos da clase JRadioButtonMenuItem compórtanse exactamente igual que os botóns de opción, polo tanto teremos que ter con eles as mesmas precaucións. Primeiramente, como queremos que unicamente un deles estea seleccionado, deberemos crear un ButtonGroup e asignarlle á propiedade buttonGroup de cada un dos JRadioButtonMenuItem o buttonGroup creado. Ademais a un deles deberemos de activarlle a súa propiedade selected de xeito que apareza marcado ao arrancar a aplicación. Unha vez modificadas as propiedades dos JRadioButtonMenuItem imos implementar os requisitos que nos piden no enunciado da aplicación. Neste caso pídennos que cando elixamos a opción Home ademais de seleccionar esa opción, amosemos unha mensaxe por pantalla indicando a selección realizada. Para a opción Muller o mesmo. Ao igual que ocorría cos botóns de opción, os JRadioButtonMenuItem lanzan o evento ItemStateChanged cando cambia o seu valor. A continuación imos ver a implementación do evento para el JRadioButtonMenuItem mnuitrbtHome:

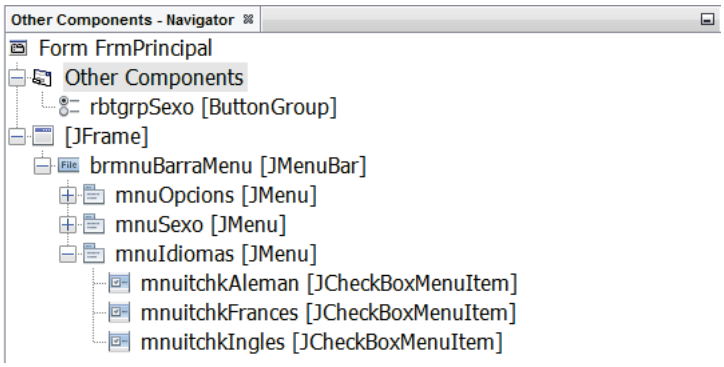
```
private void mnuitrbtHomeItemStateChanged(java.awt.event.ItemEvent evt) {
    // TODO add your handling code here:
    if(evt.getStateChange()==ItemEvent.SELECTED)
    {
        JOptionPane.showMessageDialog(this, "A opción Home foi seleccionada");
    }
}
```

Como pódese observar, o tratamento do evento é similar ao realizado para o caso dos botóns de opción.

Por ultimo, ímonos centrar na rama que colga da opción Idiomas.



Neste caso imos construír un menú cuxos compoñentes van ser JCheckBoxMenuItem. Precedemos do mesmo xeito que fixemos para os menús anteriores, pero coa precaución de engadir elementos de clase JCheckBoxMenuItem. Unha vez creados esta será a configuración do menú:



Os obxectos da clase JCheckBoxMenuItem compórtanse exactamente igual que as casillas de verificación podendo estar marcadas ou non en función do valor da súa propiedade selected. Unha vez creados os JCheckBoxMenuItem imos implementar os requisitos que nos piden no enunciado da aplicación. Neste caso pídennos que cando marquemos ou desmarquemos calquera deles amosaremos unha mensaxe por pantalla indicando a acción realizada. Ao igual que ocorría coas casillas de verificación os JCheckBoxMenuItem lanzan o evento ItemStateChanged cando cambia o seu valor. A continuación imos ver a implementación do evento para o JCheckBoxMenuItem mnuitchkAleman:

```
    private void mnuitchkAlemanItemStateChanged(java.awt.event.ItemEvent evt) {
        // TODO add your handling code here:
        if(mnuitchkAleman.isSelected())
        {
            JOptionPane.showMessageDialog(this, "A opción Alemán foi marcada");
        }
        else
        {
            JOptionPane.showMessageDialog(this, "A opción Alemán foi desmarcada");
        }
    }
```

Como se pode observar o tratamento do evento é similar ao realizado para o caso das casillas de verificación.

Un pequeno detalle que nos falta é ver como é recollido o elemento de menú marcado entre tódolos JRadioButtonMenuItem ou como e recollido o estado dos diferentes JCheckBoxMenuItem da nosa aplicación. Isto é feito na opción de menú Resumo. Os requisitos da aplicación indicaban que esta opción de menú debía amosar por pantalla o sexo seleccionado no menú sexo e os idiomas seleccionados no menú Idiomas. Para elo implementamos o seu evento actionPerformed do seguinte xeito:

```
    private void mnuitResumoActionPerformed(java.awt.event.ActionEvent evt) {
```

```
        // TODO add your handling code here:
        String sexo="SEXO: ";
        if(mnuitrbtHome.isSelected())sexo+="Home";
        else sexo+="Muller";
        String idiomas="IDIOMAS: ";
        if(mnuitchkAleman.isSelected()) idiomas+="Alemán ";
        if(mnuitchkFrances.isSelected()) idiomas+="Francés ";
        if(mnuitchkIngles.isSelected()) idiomas+="Inglés ";
        if(idiomas.compareTo("IDIOMAS: ")==0) sexo+="\nIDIOMAS: Ningún";
        else sexo+="\n"+idiomas;
        JOptionPane.showMessageDialog(this, sexo);
    }
```

Como se póde ver, o xeito de acceder aos valores establecidos sobre un JRadioButtonMenuItem é similar ao empregado para tratar cun botón de opción e o mesmo ocorre co JCheckBoxMenuItem respecto á casilla de verificación. En canto aos métodos empregados para modificar o estado do JRadioButtonMenuItem e do JCheckBoxMenuItem, tamén neste caso son os mesmos que os empregados para os botóns de opción e para as casillas de verificación respectivamente.