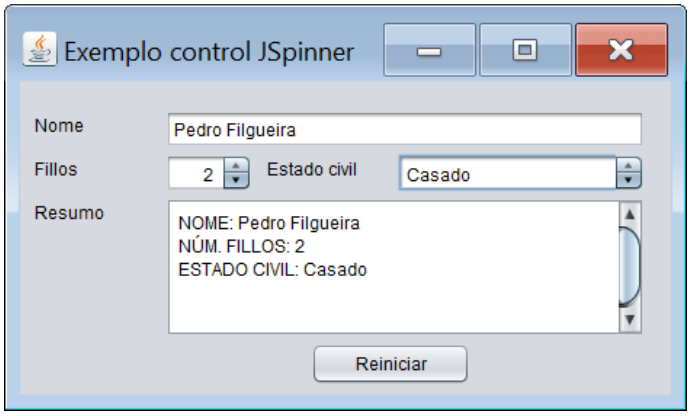
1. Compoñente Espíner (JSpinner)

O espíner é un compoñente gráfico que almacena unha serie de valores predefinidos, permitindo ao usuario seleccionar un deles. É evidente que o seu funcionamento pode ser substituído por unha lista, un combo ou incluso un grupo de botóns de opción. Non obstante, ao contrario que ocorre con estes compoñentes, co espíner non existe xeito de visualizar tódolos valores posibles que pode tomar se non é movéndonos un a un por eles. Este compoñente sóese empregar cando a secuencia entre os valores posibles que pode tomar é obvia (p.e.: 1, 2, 3 ou sota, cabalo e rei). O compoñente espíner defínese a través da clase javax.swing.JSpinner. Graficamente o seu aspecto é o seguinte:



Na imaxe anterior podemos ver o emprego de dous espíners para introducir datos nun formulario.

Propiedades do control JSpinner empregadas máis habitualmente e que pódense establecer a través do entorno de programación:

Propiedade	Función
Border	Borde do compoñente
Cursor	Mediante esta propiedade indícase a imaxe que amosará o punteiro do rato ao navegar sobre o compoñente
Enabled	Se esta propiedade está activada o compoñente será funcional. No caso de que esta propiedade estea desactivada o compoñente será visualizable, pero o usuario non poderá interaccionar con el.
Focusable	Se esta propiedade está activada o compoñente entrará na roda de reparto do foco de xeito que ao premer a tecla de cambio de foco (habitualmente o tabulador) nalgún momento tomará o foco da aplicación (cando sexa a súa quenda na roda de reparto do foco). Pola contra, se esta propiedade está desactivada o compoñente non entrará na roda de reparto do foco e soamente será posible acceder a el mediante o rato.
Font	Características da fonte do compoñente (tipo de fonte, tamaño, ...)
Model	Modelo de datos. Estrutura de datos que contén os valores posibles que pode tomar o espíner
ToolTipText	Mediante esta propiedade establécese unha mensaxe (tooltip) que será amosado ao deixar o punteiro do rato sobre o compoñente. É habitual empregar esta mensaxe para indicar cal é a utilidade do compoñente
Value	Valor seleccionado inicialmente no espíner. Debe ser un valor entre os valores posibles que pode tomar

Eventos do control JSpinner empregados máis habitualmente e que pódense establecer a través do entorno de programación:

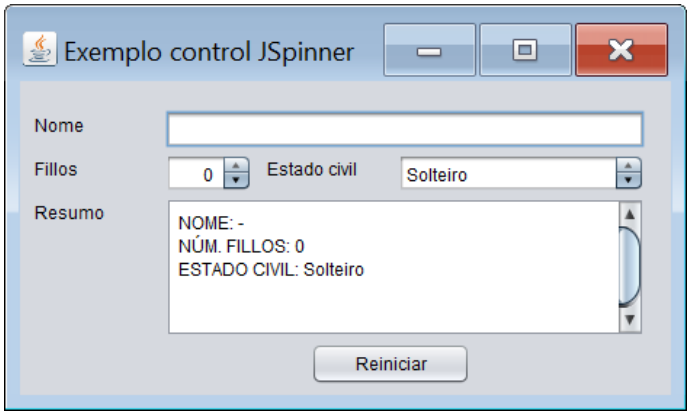
Evento	Lanzamento
StateChanged	Este evento é disparado cando cambiamos o elemento seleccionado no espíner.

Métodos do control JSpinner empregados máis habitualmente:

Método	Función
public Object getValue()	Devolve o elemento seleccionado no espíner.
public void setValue(Object value)	Establece o elemento seleccionado no espíner. Debe ser un elemento entre os valores posibles que pode tomar o compoñente.

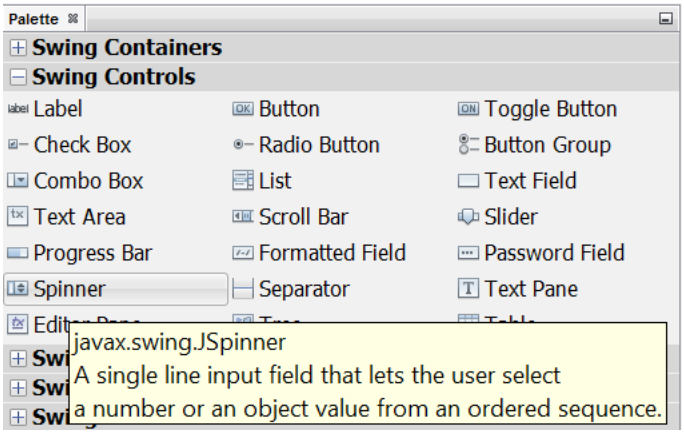
Xestión de espíners empregando NetBeans

A continuación imos desenvolver o seguinte formulario:

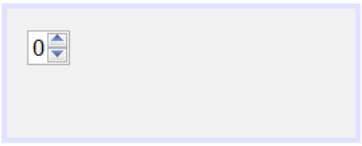


O formulario consta de dous espíners. O primeiro espíner emprégase para indicar o número de fillos que ten o usuario e pode tomar valores enteiros entre cero e vinte. O segundo espíner emprégase para indicar o estado civil do usuario. Pode tomar valores entre Solteiro, Casado e Viúvo (en realidade, neste caso sería mellor substituír o espíner por un grupo de botóns de opción, unha lista ou un combo, pero imos facelo deste xeito a fin de traballar os espíners). Ademais, cada vez que cambiemos algún valor, xa sexa na caixa de texto Nome ou a través dos espíners, modificarase o contido da área de texto Resumo a fin de amosar a situación actual dos datos introducidos no formulario. O botón Reiniciar reseteará o formulario ao seu estado inicial. Calquera erro que poida acontecer ao longo do emprego do programa debe ser reportado ao usuario.

Para engadir un compoñente de tipo JSpinner no noso formulario primeiramente seleccionarémolo da paleta de compoñentes do entorno de programación:

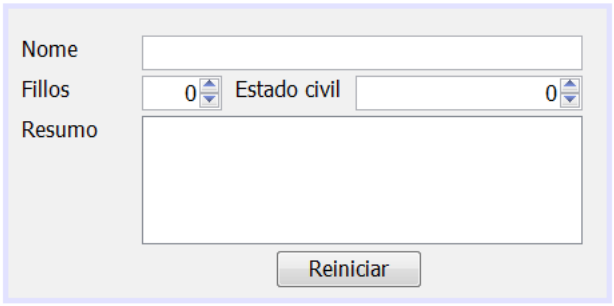


e arrastrámolo ata o formulario:

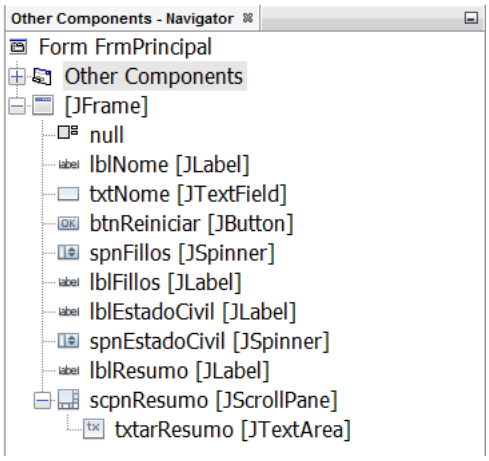


Unha vez situado el compoñente dentro do formulario, adaptámolo para que teña o aspecto visual que desexamos que teña.

No caso que estamos desenvolvendo necesitamos engadir catro etiquetas, unha caixa de texto, un botón, unha área de texto e dous espíners. Unha vez engadidos e colocados no seu lugar correspondente, este será o resultado do deseño:



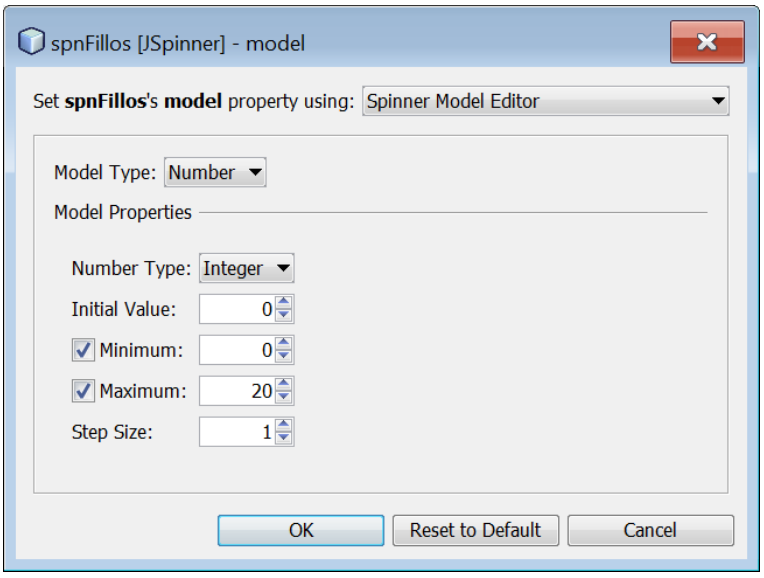
O noso formulario quedará configurado do seguinte xeito:



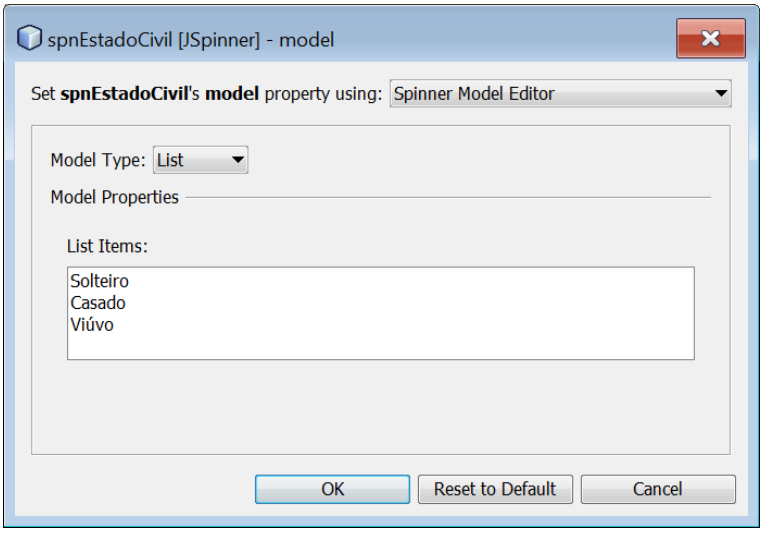
Unha vez que temos colocados os compoñentes que necesitamos hai que configurar o seu aspecto. Neste caso faremos as seguintes modificacións:

Para a área de texto txtarResumo, desmarcamos a súa propiedade editable a fin de que non sexa editable.

Respecto ao espíner spnFillos, temos que indicar o tipo de datos que almacena e cales son os valores posibles. Para elo pulsamos sobre o botón adxunto á súa propiedade model. Ao facelo, despregarase unha xanela na cal podemos configurar o modelo de datos empregado polo espíner. Nesta xanela podemos seleccionar o tipo de información que vai almacenar o espíner. Faise a través da opción model type, a cal permítenos elixir entre números, datas e listas de cadeas de texto. Neste caso queremos almacenar un número (Number) entre un e vinte, polo tanto seleccionamos números, indicamos que o tipo é número Integer e ademais definimos o valor mínimo que pode tomar o espíner (cero), o valor máximo (vinte) e en cantas unidades modificarase o valor seleccionado do espíner cada vez que interactuemos con el (unha). A xanela de configuración para a propiedade modelo do espíner spnFillos quedará do seguinte xeito:



A continuación pasamos a modificar a propiedade model do espíner spnEstadoCivil para indicar os valores posibles que pode tomar. Neste caso o tipo de información sera unha lista de cadeas de texto (List). Unicamente temos que indicar os valores de cada unha das cadeas que queremos que nos presente o espíner. A xanela de configuración para a propiedade modelo do espíner spnEstadoCivil quedará configurada do seguinte xeito:



O comportamento visual da aplicación xa está resolto. Agora imos desenvolver a súa funcionalidade.

Os requirimentos do programa pídennos que cada vez que cambiemos algún valor no formulario, modificarase o contido da área de texto Resumo a fin de amosar a situación actual dos datos introducidos no formulario. Cando realizamos algún cambio nos valores dun espíner dispararase o seu evento stateChanged. Polo tanto, co fin de cumprir os requirimentos do noso programa deberemos implementar o evento stateChanged para os dous espíners que temos na nosa aplicación. Vexamos a continuación a implementación do evento stateChanged para o espíner spnFillos:

```
    private void spnFillosStateChanged(javax.swing.event.ChangeEvent evt) {
        // TODO add your handling code here:
        txtarResumo.setText(xerarResumo());
    }
```

Ao detectar un cambio no valor seleccionado no espíner spnFillos establecemos na área de texto txtarResumo o valor devolto polo método xerarResumo, o cal encárgase de calcular o texto a amosar na área de texto txtarResumo en función dos valores dos compoñentes do formulario. Evidentemente, entre os valores devoltos polo método xerarResumo atoparase o novo valor seleccionado no espíner spnFillos. A implementación do evento stateChanged para o espíner spnEstadoCivil é similar á do espíner spnFillos.

Imos comentar como é recollido o elemento seleccionado no espíner. Esta tarefa realizámola no método xerarResumo comentado anteriormente. O xeito de capturar o valor do elemento seleccionado no espíner spnFillos é o seguinte:

```
String fillos= ((Integer)spnFillos.getValue()).toString();
```

Recordemos que cando definimos o espíner indicamos que almacenaría obxectos de tipo Integer, polo lo tanto, os obxectos que recuperamos do espíner

temos que castealos ao que realmente son, é dicir, obxectos da clase Integer. Adicionalmente estamos convertendo o valor devolto a String, pero isto xa é unicamente polos requisitos do programa.

Non obstante, o xeito de capturar o valor do elemento seleccionado no espíner spnEstadoCivil é o seguinte:

```
String estadoCivil=(String)spnEstadoCivil.getValue();
```

Neste caso, cando definimos o espíner indicamos que almacenaría obxectos de tipo cadeas de texto, polo tanto, os obxectos que recuperamos do espíner temos que castealos ao que realmente son, é dicir, obxectos de clase String.

Os requirimentos do programa tamén indícannos que ao pulsar o botón Reiniciar resetearase o formulario ao seu estado inicial. En relación cos espíners isto implica asignarlles un valor que só poderá ser un dos valores entre os posibles que poden tomar. Imos ver como é realizada esta asignación para o espíner spnFillos:

```
        spnFillos.setValue(0);
```

Simplemente temos que empregar o método setValue do espíner e pasarlle un dos valores da lista de valores que admite. Un detalle a comentar é que inicialmente o espíner spnFillos dixemos que só admitía valores de tipo Integer e a pesar diso estámoslle pasando un int. Non hai problemas debido a que o sistema realiza unha conversión de tipos automática. Converte o cero de tipo int ao cero de tipo Integer. Se quixéramos evitar esta conversión poderíamos empregar a seguinte liña para asignarlle un valor ao espíner:

```
        spnFillos.setValue(new Integer(0));
```

Respecto á asignación dun valor entre os admisibles para o espíner spnEstadoCivil, empregamos o seguinte código:

```
        spnEstadoCivil.setValue("Solteiro");
```

Ao igual que fixemos co espíner spnFillos, facendo uso do método setValue do espíner, pasámoslle o valor dun dos elementos admisibles polo espíner, neste caso o String Solteiro.