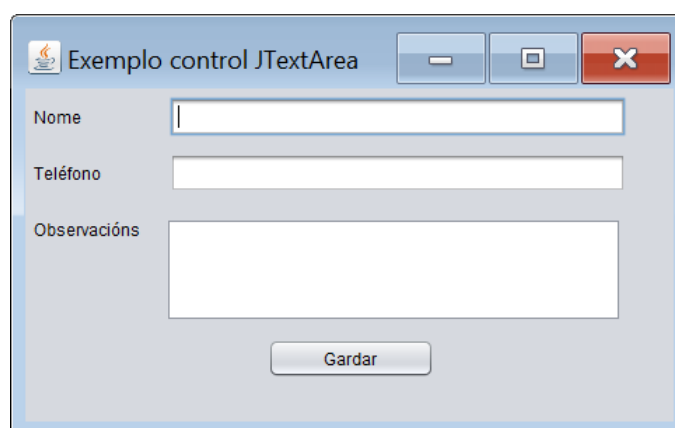


1.1.1.1 Compoñente Área de texto (JTextArea)

O comportamento do compoñente área de texto é moi similar ao do compoñente caixa de texto. Emprégase para recoller cadeas alfanuméricas introducidas polo usuario ou para amosalas. A diferenza da caixa de texto que unicamente pode mostrar unha liña, o compoñente área de texto é multiliña. É importante indicar que a capacidade para formatar o texto contido no compoñente é moi limitada (para estilizar dun xeito máis preciso un texto empréganse outros compoñentes). Tipicamente nos formularios nos que hai áreas de texto existe algún control (normalmente un botón) sobre o cal o usuario pode actuar de xeito que esa actuación desencadee a recollida dos datos das caixas de texto para o seu posterior procesamento. Aínda que non é habitual que as áreas de texto desencadeen eventos en función de accións realizadas sobre elas, cabe destacar que poden disparar eventos cada vez que realizamos algunha acción de teclado sobre elas, de xeito que podemos en todo momento realizar validacións sobre o texto que o usuario esta introducindo nun momento dado. O compoñente área de texto defínese a través da clase `javax.swing.JTextArea`. Gráficamente o seu aspecto é o seguinte:



Na imaxe anterior podemos ver un área de texto (compoñente que acompaña á etiqueta Observacións).

As áreas de texto teñen un forte compoñente de persoalización que nos permite adaptalas ás necesidades gráficas dos nosos interfaces de usuario, pero esta persoalización céntrase principalmente na persoalización do compoñente, non do texto. Para estilizar o texto este non é o compoñente axeitado. Este compoñente está orientado a texto plano.

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

Propiedade	Función
Background	Cor de fondo do compoñente
Border	Borde do compoñente
Columns	Esta propiedade establece o número de columnas visibles que conterá a área de texto. Se debido ao ancho que lle damos á área de texto algunha delas queda fora da parte visible do control, esas columnas non serán visibles. Se ademais, a área de texto está contida dentro dun <code>JScrollPane</code> , este amosará unha barra de desprazamento horizontal indicando que hai

	máis columnas na área de texto. A propiedade non limita o número de columnas que pode ter o control. Unicamente establece o seu valor inicial. Este poderá cambiar dinamicamente ao longo da execución do programa.
Cursor	Mediante esta propiedade indícase a imaxe que amosará o punteiro do rato ao navegar sobre o compoñente
DisabledTextColor	Cor do texto cando o compoñente está deshabilitado
Editable	Emprégase para indicar se podemos ou non escribir na área de texto
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, ...)
Foreground	Cor do texto do compoñente
LineWrap	Se activamos esta opción, cando unha liña non entre na parte visible do área de texto, o control partirá esa liña para continuar escribindoa nunha nova liña (wrapping)
Margin	Marxes entre os bordos do compoñente e o texto contido nel.
Rows	Esta propiedade establece o número de filas visibles que conterá o área de texto. Se debido á altura que lle damos ao área de texto algunha delas queda fora da parte visible do control, esas liñas non serán visibles. Se ademais, o área de texto está contida dentro dun JScrollPane, este mostrará unha barra de desprazamento vertical indicando que hai máis liñas no área de texto. A propiedade rows non limita o número de filas que pode ter o control. Unicamente establece o seu valor inicial. Este poderá cambiar dinamicamente ao longo da execución do programa.
SelectedTextColor	Cor do texto seleccionado na área de texto
SelectionColor	Cor de fondo do texto seleccionado na área de texto
TabSize	Mediante esta propiedade indícase o número de caracteres que ocupa unha tabulación dentro do compoñente
Text	Emprégase para establecer o texto que aparece na área de texto
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
WrapStyleWord	Esta propiedade só funcionará se a propiedade lineWrap está activa. Se wrapStyleWord está activa, cando unha palabra non entre na parte visible do área de texto, o control moverá esa palabra á seguinte liña. No caso de que a propiedade wrapStyleWord non estea activada, o wrapping (se é que activouse lineWrap) será feito a nivel de liña.

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

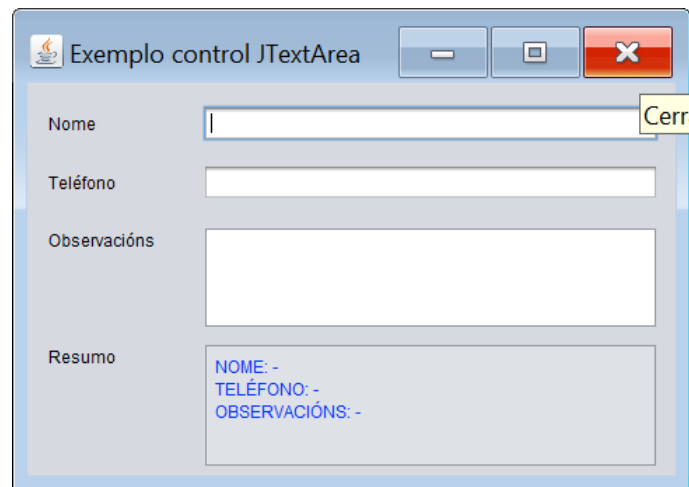
Evento	Lanzamento
KeyPressed	Este evento é disparado cando pulsamos unha tecla sobre a área de texto de texto. Se deixamos unha tecla premida, o evento dispararase tantas veces como caracteres sexan escritos na área de texto.
KeyReleased	Este evento é disparado cando deixamos de premer unha tecla sobre a área de texto. Se deixamos unha tecla premida, o evento dispararase unicamente cando deixemos de premer a tecla.
KeyTyped	Este evento é disparado cando escribimos un carácter na área de texto.

Métodos do control JTextArea empregados máis habitualmente:

Método	Función
public String getText()	Emprégase para recuperar o texto dunha área de texto
public void setText(String text)	Emprégase para establecer o texto dunha área de texto

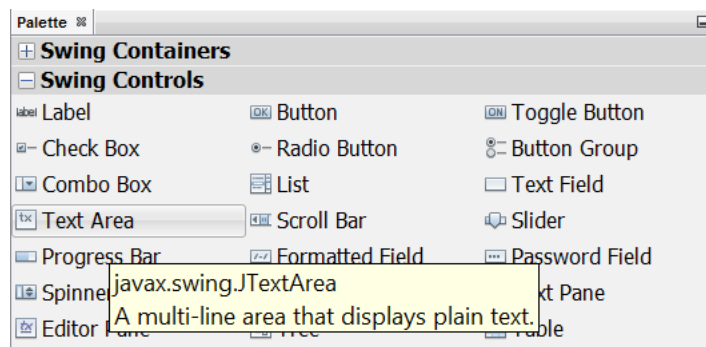
Xestión de áreas de texto empregando NetBeans

A continuación imos desenvolver o seguinte formulario:

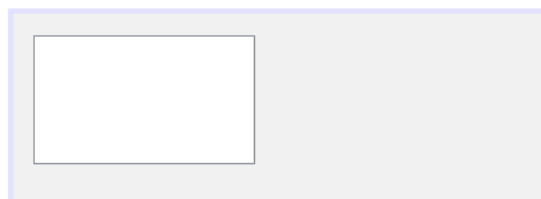


O formulario consta de dúas áreas de texto. A primeira emprégase para indicar as observacións acerca da persoa que estamos inserindo. A segunda área de texto non será editable e nela vaise ir creando dinamicamente un resumo da información que o usuario va inserindo nos diferentes campos do formulario. A caixa de texto Nome só admitirá letras, mentres que a caixa de texto Teléfono só admitirá números.

Para engadir un compoñente de tipo `JTextArea` no noso formulario primeiramente seleccionáremolo da paleta de compoñentes do entorno de programación:



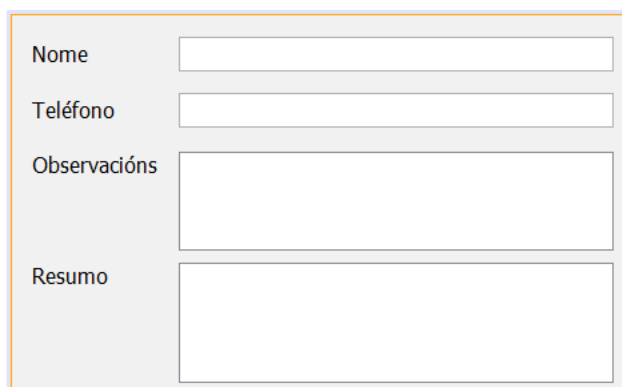
e arrastrámolo ata o formulario:



Cando arrastramos unha área de texto a noso formulario en realidade créanse dous compoñentes: un JScrollPane e dentro deste a área de texto. Isto é debido a que cando escribimos texto dentro do compoñente área de texto pode ser que a liña escrita sexa máis longa que o compoñente, o que haia máis liñas que altura ten o compoñente. Para desprazarnos polo texto poderemos empregar os cursores, aínda que é máis fácil facelo mediante barras de desprazamento. O problema é que o compoñente área de texto non ten barras de desprazamento. Aquí é onde entra en acción o entorno de programación creando o área de texto dentro dun JScrollPane, que non é máis que un panel especial que ten barras de desprazamento e que pode utilizalas para desprazar os compoñentes que contén (neste caso a área de texto contida). Como resultado, temos que aínda que a área de texto non ten barras de desprazamento, de cara ao usuario aparenta que as ten.

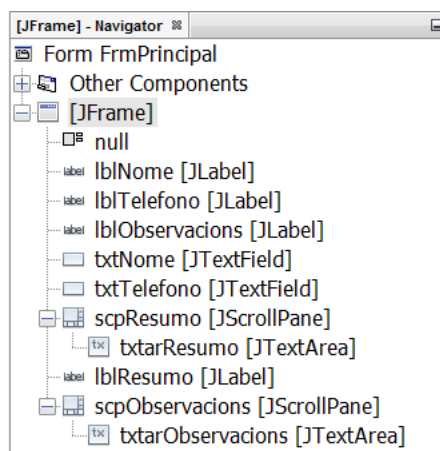
Unha vez situado o compoñente dentro do formulario, adaptámolo para que teña o aspecto visual que desexamos que teña.

No caso que estamos desenvolvendo necesitamos engadir catro etiquetas, dúas caixas de texto, dúas áreas de texto e un botón. Unha vez engadidos e colocados no seu lugar correspondente, este será o resultado do deseño:



The image shows a visual representation of a Java Swing form. It consists of a light gray rectangular container. Inside, there are four labels on the left side, each aligned with a corresponding input field on the right. The labels are 'Nome', 'Teléfono', 'Observacións', and 'Resumo'. The input fields are: a single-line text box for 'Nome', a single-line text box for 'Teléfono', a multi-line text area for 'Observacións', and a multi-line text area for 'Resumo'. The entire form is outlined with a thin orange border.

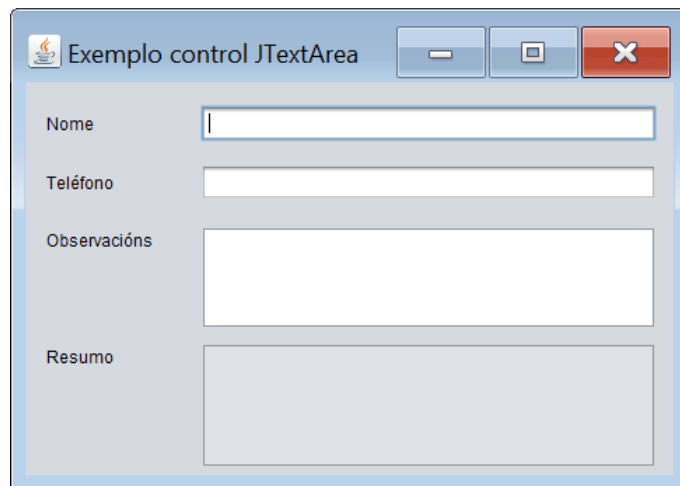
O noso formulario quedará configurado do seguinte xeito:



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

- A área de texto txtarObservaciones fará wrapping a nivel de palabra, polo tanto marcamos as súas propiedades lineWrap e wrapStyleWord
- A área de texto txtarResumo estará deshabilitada, a cor da letra será azul e fará wrapping a nivel de palabra. Para elo, desmarcamos a propiedade enabled, á propiedade DisabledTextColor asignámoslle a cor azul e marcamos as súas propiedades lineWrap e wrapStyleWord.

Se executamos a aplicación este será o resultado:



Como pódese observar o comportamento visual da aplicación xa está resolto. Agora imos desenvolver a súa funcionalidade.

A acción desenvólvese no momento que escribimos sobre calquera das caixas de texto o sobre a área de texto txtarObservacions.

Ímonos centrar en como resolvemos a acción relacionada coas áreas de texto sobre a área de texto txtarObservacions:

Cada vez que escribimos un carácter sobre a área de texto txtarObservacions, tamén temos que engadilo á área de texto txtarResumo. Para elo, implementamos o evento KeyReleased da área de texto txtarObservacions:


```
private void txtarObservacionsKeyReleased(java.awt.event.KeyEvent evt) {
    // TODO add your handling code here:
    txtarResumo.setText(xerarResumo());
}
```

Como pódese observar, cada vez que unha tecla é liberada sobre a área de texto txtarObservacions establécese o texto completo da área de texto txtarResumo empregando so seu método setText. O texto que é establecido na área de texto txtarResumo será o escrito nas dúas caixas de texto e o escrito ata o momento na área de texto txtarObservacions, concatenado cunha serie de descrições para cada un deles. Para calcular este texto empregamos o método xerarResumo (ver código fonte da aplicación). Do método xerarResumo cabe destacar a seguinte liña:

```
String observacions=txtarObservacions.getText().trim();
```

A liña de código amosada emprégase para recoller o contido da área de texto txtarObservacions. Como pódese observar, este proceso é realizado aplicando sobre a área de texto o seu método getText e gardando o valor devolto nunha variables de tipo String.

Se executamos a aplicación introducindo datos, este será o resultado da execución:



The screenshot shows a Java Swing window titled "Exemplo control JTextArea". The window contains a form with the following fields and content:

- Nome:** Marta Enes
- Teléfono:** 981111111
- Observacións:** Só disponible martes e xoves
Mellor chamar polas tardes de 18.00 a 20.00
- Resumo:** NOME: Marta Enes
TELÉFONO: 981111111
OBSERVACIÓNS: Só disponible martes e xoves
Mellor chamar polas tardes de 18.00 a 20.00

The text in the "Resumo" section is displayed in blue, indicating it might be a default or placeholder text.