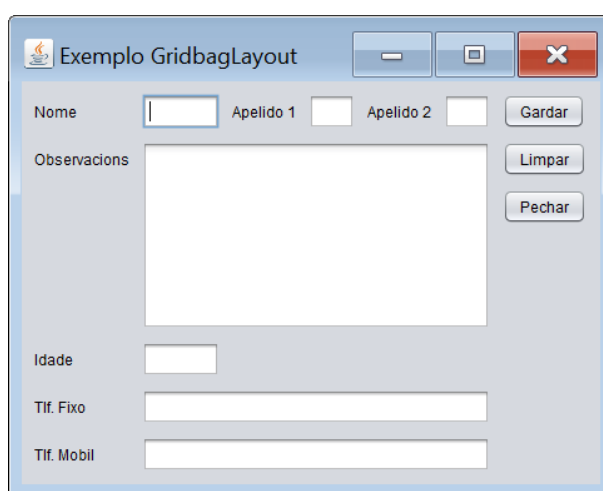


1.1.1.1 Layout manager GridBagLayout

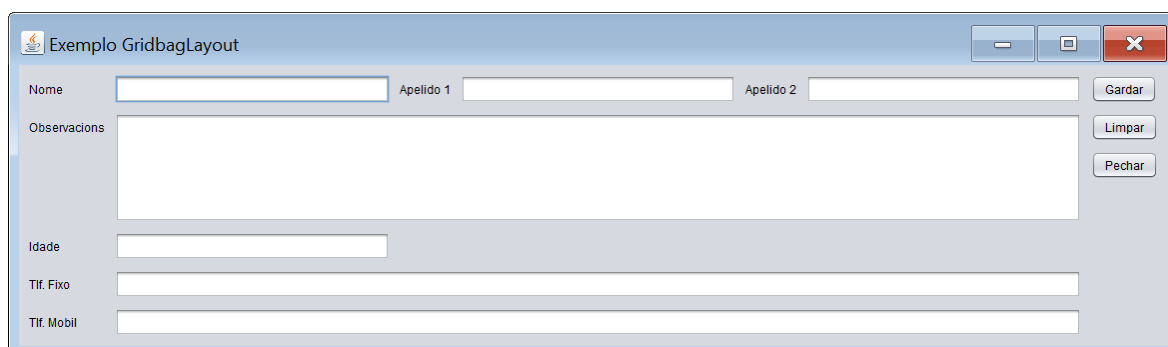
O xestor de distribución `GridBagLayout` é o layout manager que nos permite desenvolver as distribucións de compoñentes máis complexas. É con diferenza o xestor de distribución máis completo e flexible. Non obstante, esta maior funcionalidade supón un aumento na complexidade de seu emprego.

O layout manager `GridBagLayout` é un layout manager `GridLayout` ampliado. No layout manager `GridLayout`, o espazo divídese nunha malla de celas de tamaño similar. Porén, no layout manager `GridBagLayout` cada cela pode ter diferente tamaño e ademais un compoñente pode estenderse ao longo de diversas celas contiguas (tanto horizontal como verticalmente).

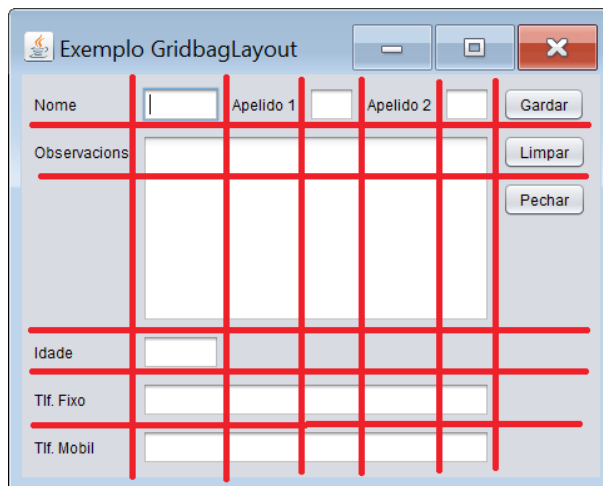
O layout manager `GridBagLayout` defínese a través da clase `java.awt.GridBagLayout`. O seu aspecto visual é o seguinte:



O mesmo contedor redimensionado:



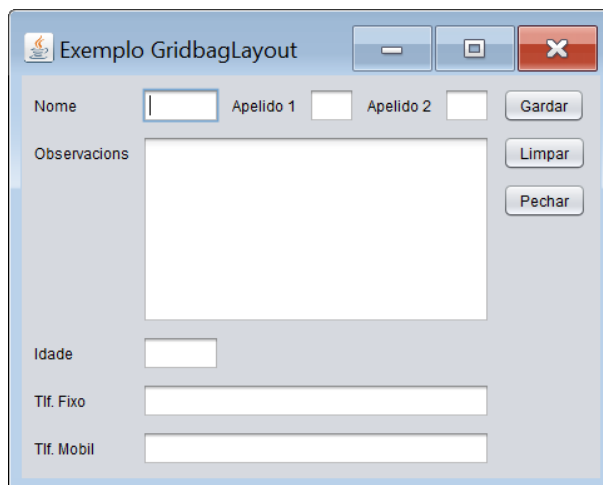
Como se pode observar, independentemente de como están definidos os diferentes compoñentes do contedor, hai unha estrutura de celas que se mantén sen importar o tamaño do contedor. Na seguinte imaxe pódese ver esta estrutura de celas:



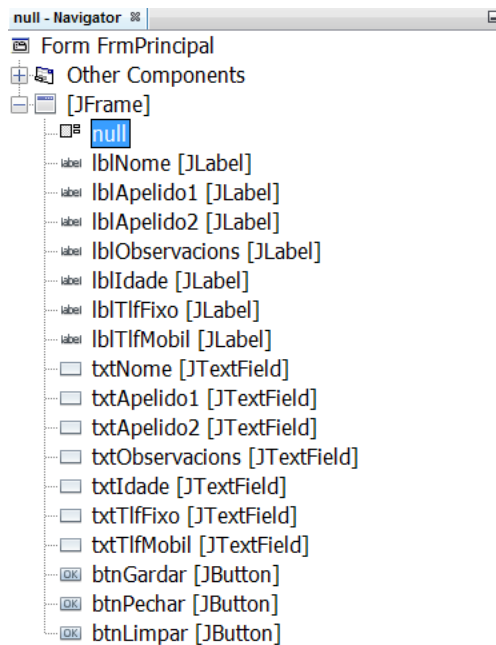
Para acadar os resultados desexados empregando este layout manager, débese xogar coas propiedades dos compoñentes que forman parte do mesmo.

Configuración dun GridBagLayout empregando NetBeans

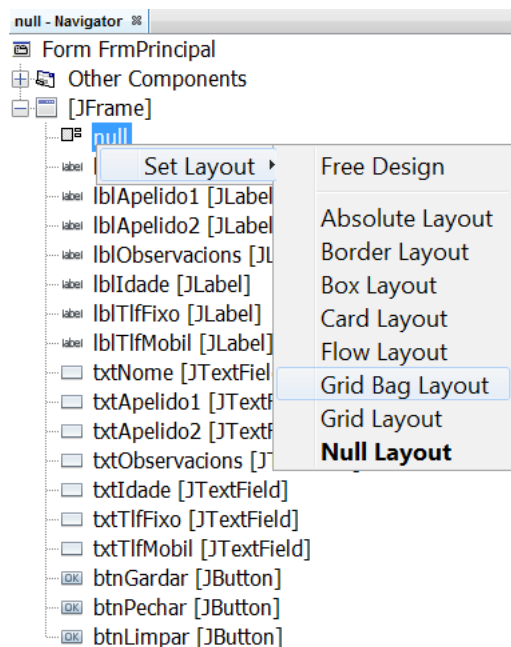
A continuación imos desenvolver o seguinte formulario:



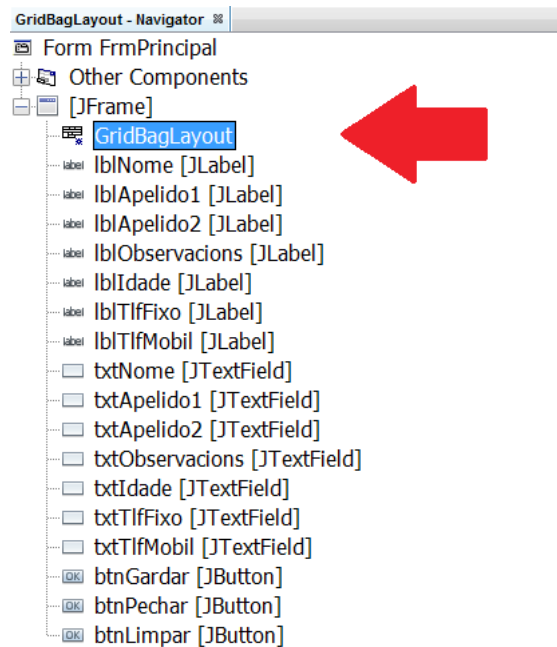
Para empezar, creamos un proxecto e engadimos nel un formulario JFrame. Dentro do noso formulario engadimos sete etiquetas, sete caixas de texto e tres botóns:



O seguinte que debemos facer é indicar que queremos establecer un layout manager de tipo GridBagLayout sobre o noso formulario. Para elo prememos sobre JFrame co botón dereito, eliximos a opción Set Layout e seleccionamos Grid Bag Layout.



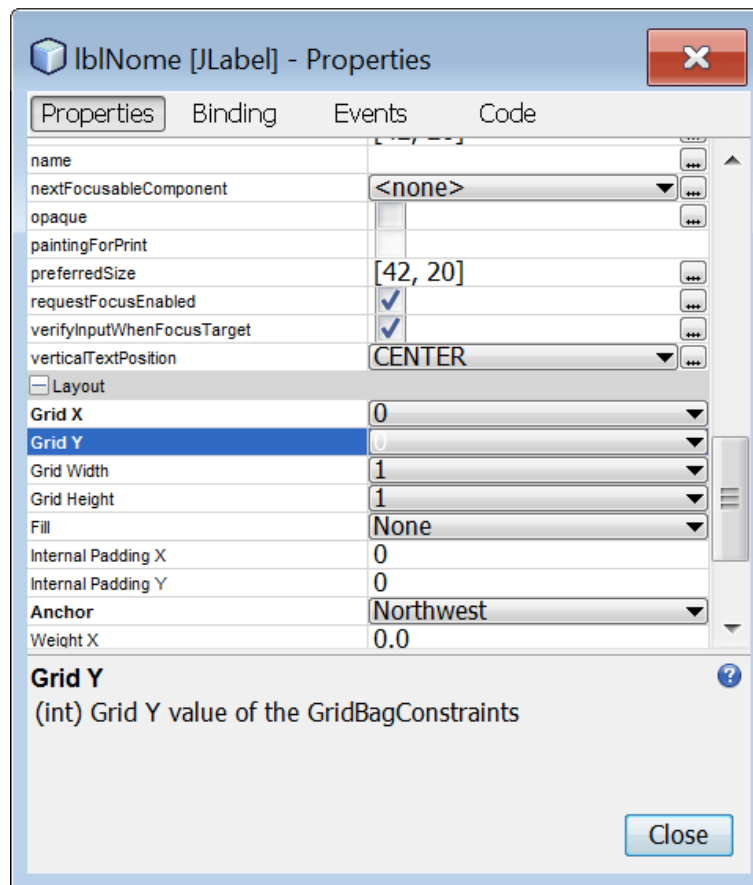
A partir deste momento temos establecido para o noso formulario como layout manager un layout manager GridBagLayout:



Se executamos nosa aplicación veremos que o resultado non se parece nada ao buscado. O primeiro que debemos facer é definir a configuración da matriz de celas do noso formulario. Esta matriz non é máis que un modelo teórico. No exemplo que estamos desenvolvendo, a forma da matriz é a seguinte:

	0	1	2	3	4	5	6	FILA
0	Nome	<input type="text"/>	Apelido 1	<input type="text"/>	Apelido 2	<input type="text"/>	Gardar	
1	Observacions	<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="text"/>	Limpar	
2		<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="text"/>	Pegar	
3	Idade	<input type="text"/>						
4	Tlf. Fixo	<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="text"/>		
5	Tlf. Mobil	<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="text"/>		
COLUMNA								

Ao definir esta matriz hai que ter en conta unha regra básica que é a seguinte: un compoñente pode ocupar máis dunha cela da matriz, pero nunca dous o máis compoñentes deben ocupar as mesmas celas da matriz (en realidade poden, pero complica o deseño do xestor de distribución innecesariamente). Unha vez que temos definida teoricamente a matriz de compoñentes que buscamos pasamos a implementar o deseño teórico sobre o entorno de programación. A implementación da matriz teórica faise a través das propiedades Grid X e Grid Y de cada un dos compoñentes que forman parte da matriz. Mediante estas propiedades definimos a posición esquerda superior que ocupa o compoñente na matriz de celas:



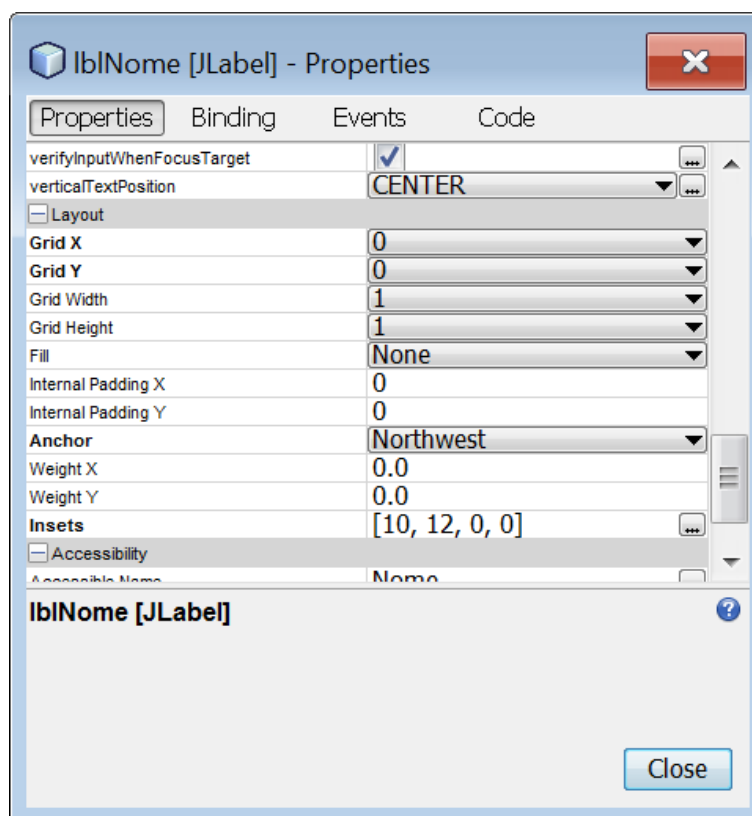
No caso que estamos desenvolvendo estes son os valores para cada un dos compoñentes:

Compoñente	Grid X	Grid Y
IblNome	0	0
IblApellido1	2	0
IblApellido2	4	0
IblObservacions	0	1
IblIdade	0	3
IblTlfFixo	0	4
IblTlfMobil	0	5
txtNome	1	0
txtApellido1	3	0
txtApellido2	5	0
txtObservacions	1	1
txtIdade	1	3
txtTlfFixo	1	4
txtTlfMobil	1	5
btnGardar	6	0
btnPechar	6	1
btnLimpar	6	2

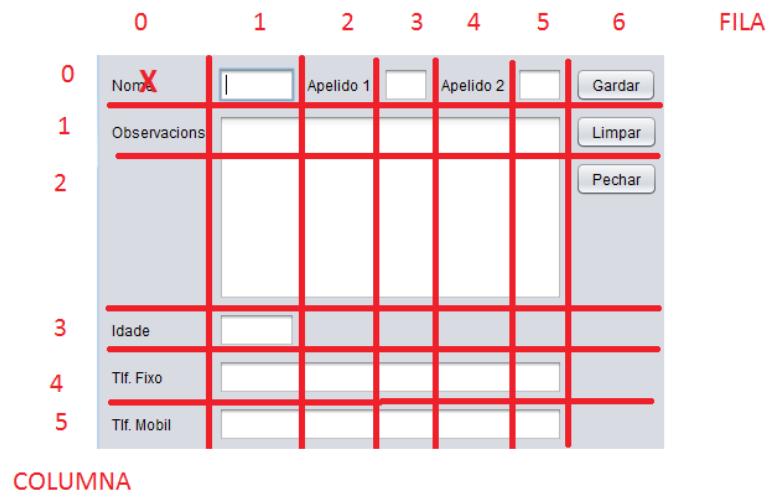
Se executamos a nosa aplicación veremos que o resultado continua sen parecerse ao buscado, aínda que agora, a estrutura vaise asemellando lixeiramente á composición buscada.



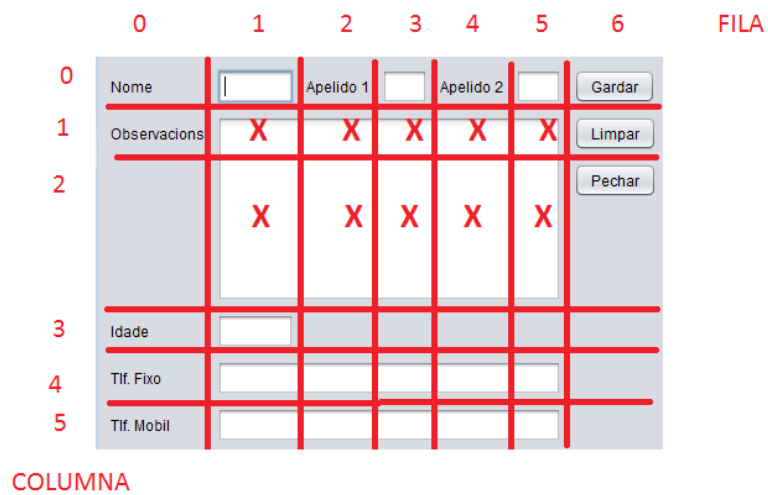
O seguinte que debemos facer é establecer cantas celas ocupa cada compoñente. Para definir cantas celas ocupa cada compoñente facémolo a través das propiedades Grid Width e Grid Height. Mediante Grid Width indicamos cantas celas tomamos ao ancho da matriz de celas e mediante Grid Height indicamos cantas celas tomamos ao alto da matriz de celas. O número total de celas que utiliza un compoñente ven dado por $\text{Grid Width} * \text{Grid Height}$ do compoñente.



Por exemplo, para a etiqueta lblNobre, Grid Width vale 1 e Grid Height vale 1, xa que só empregamos unha cela para debuxar ese compoñente.



Pero para a caixa de texto txtObservacions, Grid Width vale 5 e Grid Height vale 2, xa que empregamos cinco columnas e dous filas para debuxar ese compoñente.



No caso que estamos desenvolvendo estes son os valores para cada un dos compoñentes:

Compoñente	Grid Width	Grid Height
lblNome	1	1
lblApellido1	1	1
lblApellido2	1	1
lblObservacions	1	1
lblIdade	1	1
lblTifFixo	1	1
lblTifMobil	1	1
txtNome	1	1
txtApellido1	1	1
txtApellido2	1	1
txtObservacions	5	2
txtIdade	1	1

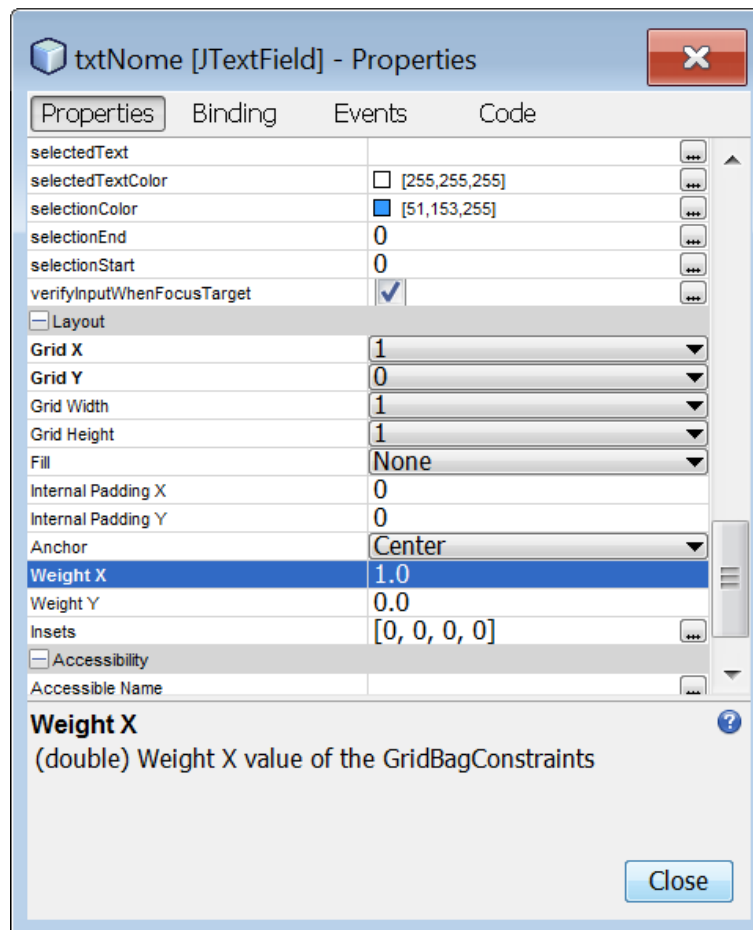
txtTlfFixo	5	1
txtTlfMobil	5	1
btnGardar	1	1
btnPechar	1	1
btnLimpar	1	1

Se executamos a nosa aplicación veremos que o resultado apenas varía respecto á última execución:



Como se pode observar, hai moito espazo desperdiciado no contedor. Isto é debido a que aínda que xa indicamos que celas están ocupadas por que compoñentes, aínda non temos establecido cales son os compoñentes que poden modificar o seu tamaño e con elo aproveitar o espazo sobrante no contedor. Por defecto ningún compoñente pode modificar o seu tamaño e polo tanto ocorre o que vemos na imaxe anterior. Para indicar que compoñentes poden modificar o seu tamaño para aproveitar o espazo restante do contedor empréganse as propiedades Weight X e Weight Y dos compoñentes. Ambas as dúas propiedades poden tomar un valor decimal, aínda que os valores habituais son 0.0 e 1.0.

- Con Weight X valendo 1.0 indicamos que o compoñente para o que aplicamos a propiedade pode axustar o seu tamaño horizontal para aproveitar o espazo sobrante do contedor. Non obstante, se dámoslle o valor 0.0 indicamos que o compoñente non pode modificar o seu tamaño horizontal para axustarse ás modificacións de tamaño do contedor.
- Con Weight Y valendo 1.0 indicamos que o compoñente para o que aplicamos a propiedade pode axustar o seu tamaño vertical para aproveitar o espazo sobrante do contedor. Non obstante, se dámoslle o valor 0.0 indicamos que o compoñente non pode modificar o seu tamaño vertical para xustarse ás modificacións de tamaño do contedor.

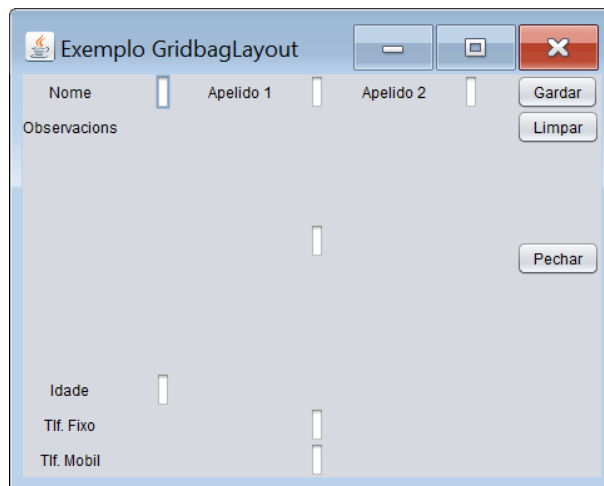


No caso que estamos desenvolvendo estes son os valores para cada un dos compoñentes:

Compoñente	Weight X	Weight Y
lblNome	0.0	0.0
lblApellido1	0.0	0.0
lblApellido2	0.0	0.0
lblObservacions	0.0	0.0
lblIdade	0.0	0.0
lblTlfFixo	0.0	0.0
lblTlfMobil	0.0	0.0
txtNome	1.0	0.0
txtApellido1	1.0	0.0
txtApellido2	1.0	0.0
txtObservacions	1.0	1.0
txtIdade	1.0	0.0
txtTlfFixo	1.0	0.0
txtTlfMobil	1.0	0.0
btnGardar	0.0	0.0
btnPechar	0.0	0.0

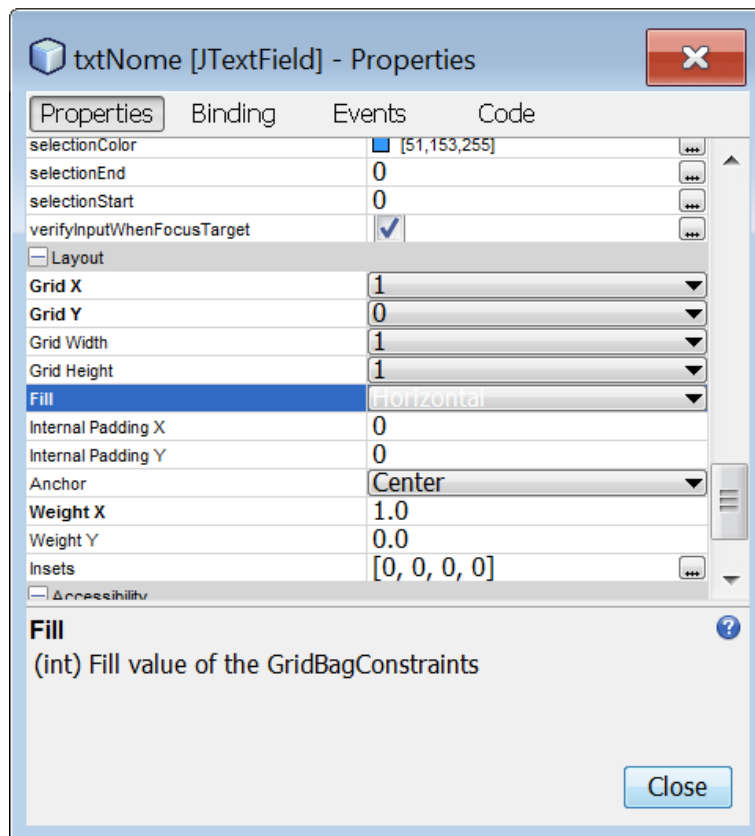
btnLimpar	0.0	0.0
-----------	-----	-----

Se executamos a nosa aplicación veremos que o resultado apenas volveu variar respecto á última execución:



Isto é debido a que coas propiedades Weight X e Weight indicamos que compoñentes poden crecer ou decrecer en función do tamaño do contedor, pero fainos falla dicir que compoñentes deben modificar o seu tamaño e en que dirección. Para establecer esta característica emprégase a propiedade Fill. Esta propiedade pode tomar os seguintes valores: None, Horizontal, Vertical e Both.

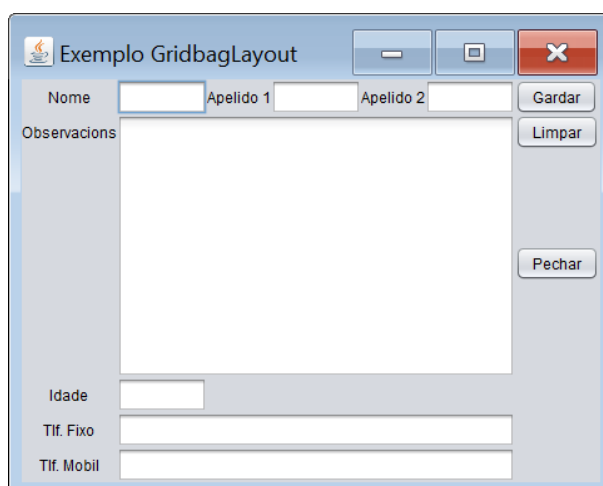
- Co valor None indicamos que o compoñente non modificará o seu tamaño
- Co valor Horizontal indicamos o compoñente modificará o seu tamaño horizontalmente dentro das celas que ocupa (sempre e cando a súa propiedade Weight X permítalo).
- Co valor Vertical indicamos que o compoñente modificará o seu tamaño verticalmente dentro das celas que ocupa (sempre e cando a súa propiedade Weight Y permítalo).
- Co valor Both indicamos que o compoñente modificará o seu tamaño horizontal e verticalmente dentro das celas que ocupa (sempre e cando as súas propiedades Weight X e Weight Y permítano).



No caso que estamos desenvolvendo estes son os valores para cada un dos compoñentes:

Compoñente	Fill
lblNome	None
lblApellido1	None
lblApellido2	None
lblObservacions	None
lblIdade	None
lblTlfFixo	None
lblTlfMobil	None
txtNome	Horizontal
txtApellido1	Horizontal
txtApellido2	Horizontal
txtObservacions	Both
txtIdade	Horizontal
txtTlfFixo	Horizontal
txtTlfMobil	Horizontal
btnGardar	None
btnPechar	None
btnLimpar	None

Se executamos a nosa aplicación veremos que agora o resultado xa é bastante similar ao buscado:



O último que nos falta por facer é aliar os compoñentes axeitadamente. Esta tarefa realízase a través da propiedade `Anchor` de cada compoñente. Esta propiedade pode tomar diferentes valores que serven para indicar onde se aliará o compoñente dentro da súa cela. Os valores empregados máis habitualmente son os seguintes:

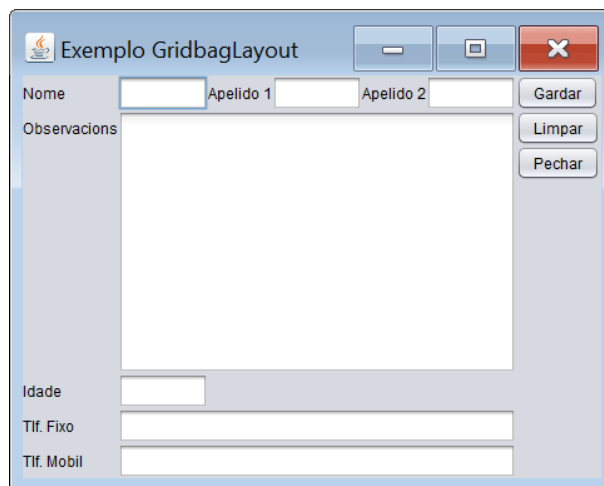
- **North:** alia o compoñente co centro do borde superior da cela.
- **South:** alia o compoñente co centro do borde inferior da cela.
- **West:** alia o compoñente co centro do borde esquerdo da cela.
- **East:** alia o compoñente co centro do borde dereito da cela.
- **Northwest:** alia o compoñente co borde superior esquerdo da cela.
- **Northeast:** alia o compoñente co borde superior dereito da cela.
- **Southwest:** alia o compoñente co borde inferior esquerdo da cela.
- **Northeast:** alia o compoñente co borde inferior dereito da cela.
- **Center:** centra o compoñente na cela.

No caso que estamos desenvolvendo estes son os valores para cada un dos compoñentes:

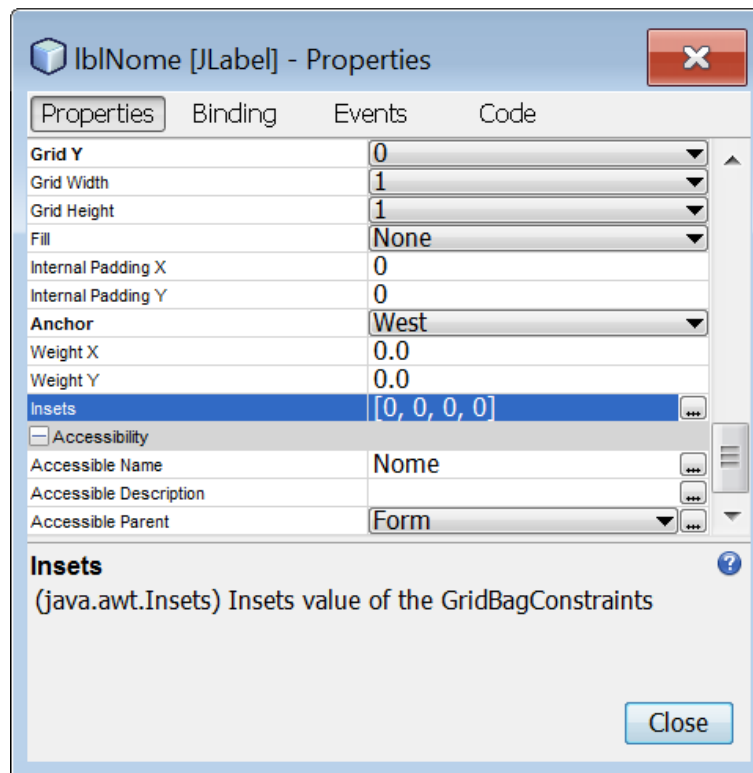
Compoñente	Anchor
lblNome	West
lblApelido1	West
lblApelido2	West
lblObservacions	West
lblIdade	West
lblTifFixo	West
lblTifMobil	West
txtNome	West
txtApelido1	West
txtApelido2	West

txtObservacions	West
txtIdade	West
txtTlfFixo	West
txtTlfMobil	West
btnGardar	North
btnPechar	North
btnLimpar	North

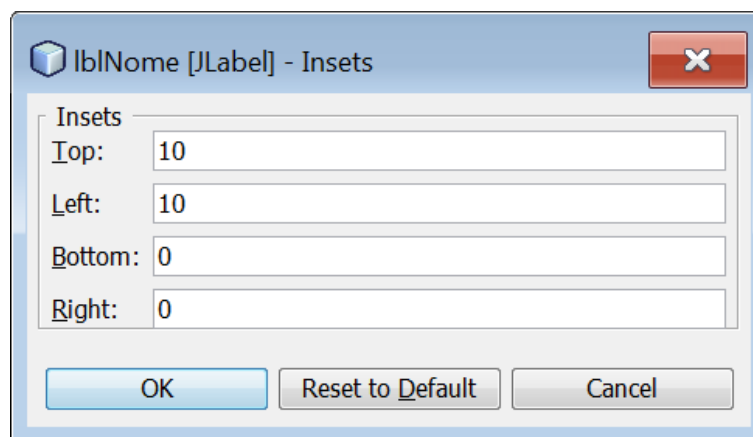
Se executamos a nosa aplicación vemos que agora o resultado é o seguinte:



Fixándonos, caeremos na conta de que os compoñentes do contedor atópanse demasiado pegados uns aos outros. Para establecer unha separación entre un compoñente e os seus adxacentes modificamos a súa propiedade Insets. Para elo, accedemos ás propiedades do compoñente de xeito que accedemos á seguinte pantalla:



Nesta pantalla pulsamos sobre o botón asociado á propiedade Insets de xeito que accedemos a esta outra pantalla:



É nesta pantalla onde establecemos o valor da separación que vai ter o noso compoñente con calquera outro compoñente adxacente.

- A través de Top establecemos a separación en píxels entre o noso compoñente e o compoñente que atópase sobre el.
- A través de Left establecemos a separación en píxels entre o noso compoñente e o compoñente que atópase á súa esquerda.
- A través de Bottom establecemos a separación en píxels entre o noso compoñente e o compoñente que atópase baixo el.
- A través de Right establecemos a separación en píxels entre o noso compoñente e o compoñente que atópase á súa dereita.

No caso que estamos desenvolvendo estes son os valores para cada un dos compoñentes:

Compoñente	Insets (Top)	Insets (Left)	Insets (Bottom)	Insets (Right)
lblNome	10	10	0	0
lblApelido1	10	10	0	0
lblApelido2	10	10	0	0
lblObservacions	10	10	0	0
lblIdade	10	10	0	0
lblTlfFixo	10	10	0	0
lblTlfMobil	10	10	10	0
txtNome	10	10	0	0
txtApelido1	10	10	0	0
txtApelido2	10	10	0	0
txtObservacions	10	10	0	0
txtIdade	10	10	0	0
txtTlfFixo	10	10	0	0
txtTlfMobil	10	10	10	0
btnGardar	10	10	0	10
btnPechar	10	10	0	10
btnLimpar	10	10	0	10

Se agora executamos a nosa aplicación podemos comprobar que conseguimos o resultado buscado:

The screenshot shows a Java Swing window titled "Exemplo GridbagLayout". The window has a standard title bar with minimize, maximize, and close buttons. The main content area contains a form with the following elements:

- Labels: "Nome", "Apelido 1", "Apelido 2", "Observacions", "Idade", "Tlf. Fixo", and "Tlf. Mobil".
- Text Fields: There are text input fields for "Nome", "Apelido 1", "Apelido 2", "Idade", "Tlf. Fixo", and "Tlf. Mobil".
- Buttons: There are three buttons labeled "Gardar", "Limpar", and "Pechar" located on the right side of the form.
- Text Area: A large text area for "Observacions" is located in the center of the form.

No caso de que redimensionemos o contedor, os seus compoñentes axustaranse ás novas medidas, pero sempre o farán respectando a estrutura fixada a través do deseño do noso layout manager GridBagLayout:

Comentarios adicionais sobre as propiedades Weight X e Weight Y

Imos desenvolver este comentario sobre a propiedade Weight X, pero todo o que vai ser explicado é igualmente aplicable á propiedade Weight Y pero tendo en conta que se aplicará sobre un despregue de compoñentes en vertical.

Ben, anteriormente comentamos que a propiedade Weight X tipicamente vale 0.0 (o compoñente non pode modificar o seu tamaño horizontal para adecuarse ás modificacións de tamaño do contedor) ou 1.0 (o compoñente pode adecuar o seu tamaño horizontal para aproveitar o espazo sobrance do contedor). Pero, ¿que ocorre se empregamos outros valores?. Nese caso o que ocorre é algo parecido ao que ocorría no layout manager BoxLayout cando non había espazo suficiente para aplicar o seu `maximumSize`. Lembremos que neste caso facíase un reparto proporcional do espazo. Aquí ocorre algo similar. Vexámolo sobre o exemplo que estamos empregando para desenvolver o layout manager GridBagLayout. Supoñamos que modificamos nos seguintes compoñentes o valor da propiedade Weight X:

Compoñente	Weight X
txtNome	10
txtApelido1	5
txtApelido2	5

É recomendable empregar números enteiros aínda que é posible usar decimais. Se modificamos as propiedades dos compoñentes tal e como temos establecido na táboa anterior este é o resultado:

¿Que ocorre se redimensionamos o contedor?. Pois ocorre o seguinte:

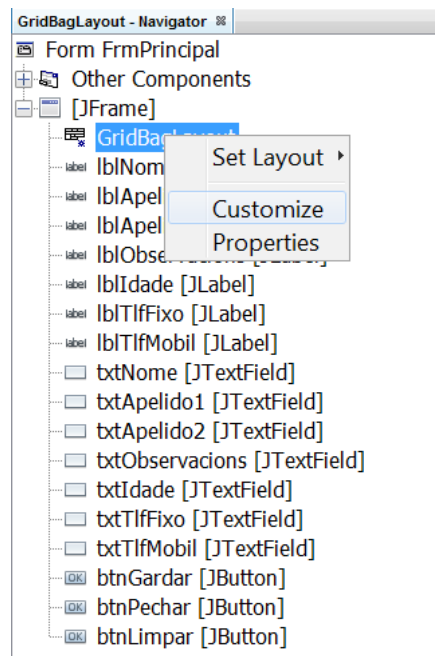
Como se pode observar, a primeira caixa de texto, sempre é o dobre de grande que as outras dúas e ademais a segunda caixa de texto e a terceira sempre son iguais. Isto é debido aos valores que temos establecido para as súas propiedades Weight X: 10, 5 e 5. Se sumamos os valores das propiedades Weight X de tódalas caixas de texto da fila (nome, apelido 1 e apelido 2) obtemos un total de 20. O espazo sobrannte unha vez pintado cada un dos compoñentes da fila repártese proporcionalmente entre cada compoñente en función do valor da súa propiedade Weight X. Neste caso:

Compoñente da fila	Weight X	% de espazo sobrannte que correspóndelle unha vez debuxados tódolos compoñentes da fila
blNome	0	0
blApelido1	0	0
blApelido2	0	0
txtNome	10	50
txtApelido1	5	25
txtApelido2	5	25
btnGardar	0	0

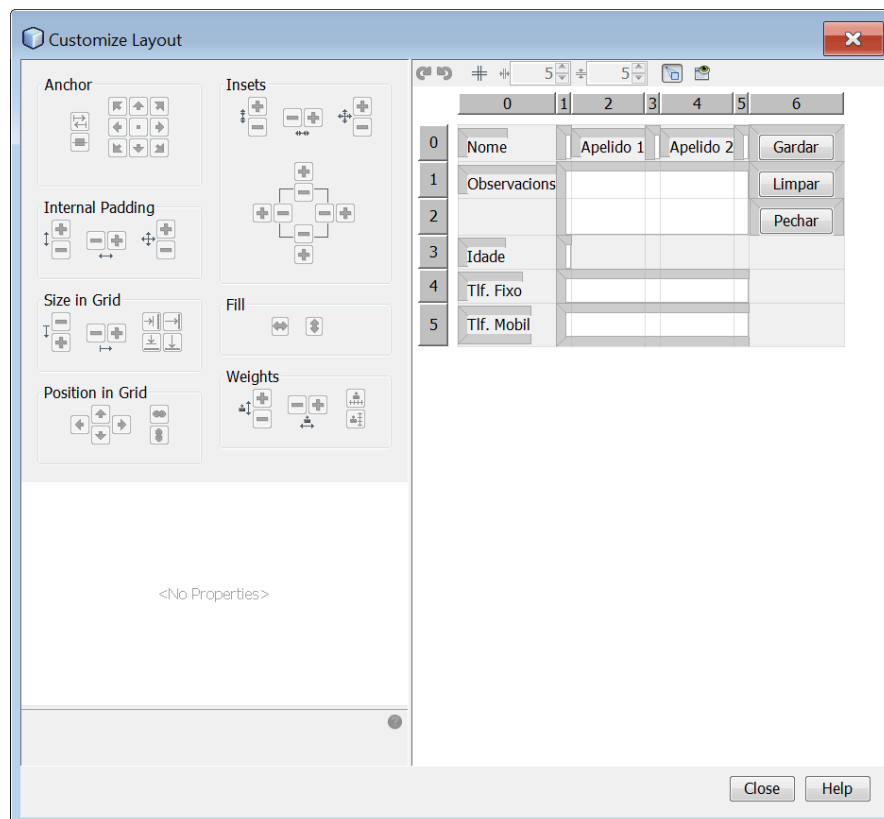
O que fai o layout manager GridBagLayout é distribuír o espazo sobrannte entre os compoñentes proporcionalmente á súa propiedade Weight X (p.e., neste caso o total de Weight X das caixas de texto da fila é 20. A txtNome correspóndelle o 50% do espazo sobrannte xa que o seu Weight X é 10, é dicir, o 50% do total. A txtApelido1 correspóndelle o 25% do espazo sobrannte xa que o seu Weight X é 5, é dicir, o 25% do total e a txtApelido2 tamén correspóndelle o 25% do espazo sobrannte xa que o seu Weight X é 5, é dicir, o 25% do total).

Emprego do asistente

NetBeans proporciona un asistente para configurar o layout manager GridBagLayout. Para empregalo unicamente debemos premer co botón dereito sobre o GridBagLayout do contedor e elixir a opción Customize do menú emerxente despregado:



Abrirásenos un asistente no cal podemos configurar graficamente o noso layout manager GridBagLayout:



Aínda que inicialmente é máis cómodo empregar o asistente para deseñar os nosos layout managers GridBagLayout, as veces, como ocorre en xeral con tódolos métodos de xeración de código automático, fai cousas que non debe e que despois son difíciles de arranxar se non se comprende exactamente para que serve cada propiedade do layout manager GridBagLayout.