1. Compoñente Botón de opción (JRadioButton)

Cando nun programa os datos de entrada que solicitamos ao usuario son valores que unicamente poden ter un valor de varios posibles, o control máis axeitado para recuperar a información é o compoñente de tipo botón de opción. Poderíase empregar outro compoñente como por exemplo unha caixa de texto, pero sen lugar a dúbidas fai que a interface sexa menos clara, complica o seu manexo de cara ao usuario e fai a súa programación máis complexa. O control botón de opción nunca vai só, senón que en realidade está composto por unha agrupación de dous ou máis botóns de opción, onde cada un deles representa unha das opcións posibles que pode tomar o usuario. Finalmente o usuario en función da súa decisión seleccionará un dos botóns de opción que compoñen a agrupación. Es tarefa do programador o asignar a selección realizada polo usuario a unha representación lóxica interna no programa. Habitualmente un botón de opción vai acompañado dun texto (provisto polo propio compoñente) que se emprega para indicar ao usuario do interface gráfico cal é a utilidade do compoñente. Os botóns de opción ademais de empregarse para tomar datos de entrada, tamén poden empregarse para amosar o estado dalgún elemento durante a execución do noso programa.

Tipicamente, nos formularios nos que hai botóns de opción existe algún control (normalmente un botón) sobre o cal o usuario pode actuar de xeito que esa actuación desencadee a recollida das agrupacións de botóns de opción para o seu posterior procesamento. En moitas ocasións o actuar sobre un botón de opción implica o desencadeamento de eventos que cambian o aspecto da interface de usuario (p.e. Imaxinemos que temos unha aplicación para crear documentos e esta aplicación ten unha agrupación con dous botóns de opción: DNI e pasaportes. Ao marcar o botón de opción DNI habilítanse os compoñentes necesarios para a toma de datos para a creación do DNI, mentres que ao marcar o botón de opción pasaporte habilítanse os compoñentes necesarios para a toma de datos para a creación do pasaporte. Tamén cabe mencionar que podemos cambiar o aspecto do control para que en lugar de amosarse un círculo que pode ser ou non marcado, amósense diferentes imaxes (acompañadas ou non de texto) en función do estado do botón de opción.

O compoñente botón de opción defínese a través da clase javax.swing.JRadioButton. Graficamente o seu aspecto é o seguinte:

Na imaxe anterior podemos ver seis botóns de opción acompañados por outros controis. Os tres primeiros están situados dentro do panel Material e presentan o aspecto habitual do compoñente botón de opción. Os outros tres encóntranse agrupados dentro do panel Cor e amosan unha imaxe persoalizada para cada un deles.

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

Propiedade	Función
Background	Cor de fondo do compoñente
ButtonGroup	Emprégase para indicar a que grupo de botóns de opción pertence este botón. Un grupo de botóns de radio permite que unicamente un dos botóns de radio dos que pertencen ao grupo poida estar activo.
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, ...)
Foreground	Cor do texto do compoñente
HorizontalTextPosition	Localización horizontal do texto do compoñente respecto á imaxe amosada polo compoñente
Icon	Mediante esta propiedade indícase a imaxe a amosar no compoñente. Se non indicamos nada o compoñente amosará o seu aspecto estándar.
IconTextGap	Separación entre o texto do compoñente e a imaxe empregada polo compoñente
PressedIcon	Mediante esta propiedade indícase a imaxe a amosar cando prememos sobre o compoñente.
RollOverEnabled	Se esta propiedade está desactivada, cando pasemos sobre o compoñente amosarase a imaxe indicada na súa propiedade RollOverIcon. Se RollOverIcon non está definida non se amosará ningunha imaxe. Pola contra, se a propiedade RollOverEnabled está activada, cando pasemos sobre o compoñente amosarase a imaxe indicada na súa propiedade RollOverIcon. Se RollOverIcon non está definida amosarase a imaxe definida na súa propiedade SelectedIcon e se a propiedade SelectedIcon non está definida non se amosará nada.
RollOverIcon	Mediante esta propiedade indícase a imaxe a amosar cando pasamos co rato sobre o compoñente.
Selected	Indica se o botón de opción esta marcado ou non
SelectedIcon	Mediante esta propiedade indícase a imaxe a amosar cando o compoñente está seleccionado.

Text	Emprégase para establecer o texto que aparece no botón de opción
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
VerticalTextPosition	Localización vertical do texto do compoñente respecto á imaxe amosada polo compoñente

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

Evento	Lanzamento
ItemStateChanged	Este evento é disparado cando cambiamos o estado do botón de opción, xa sexa por acción directa do usuario ou a través do código.

Métodos do control JRadioButton empregados máis habitualmente:

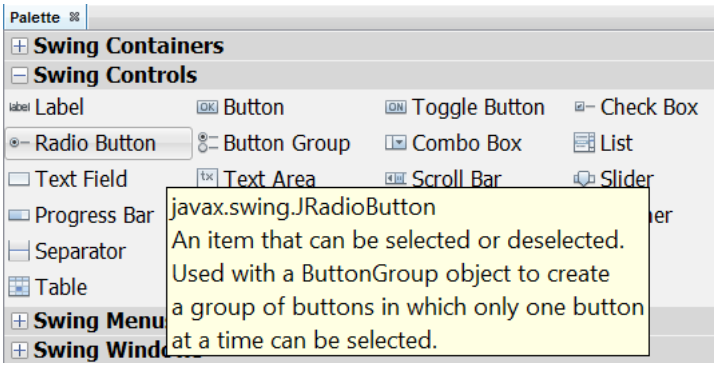
Método	Función
public boolean isSelected()	Emprégase para recuperar o estado do botón de opción
public void setSelected(boolean b)	Emprégase para establecer o valor do botón de opción

Xestión de botóns de opción empregando NetBeans (primeiro exemplo)

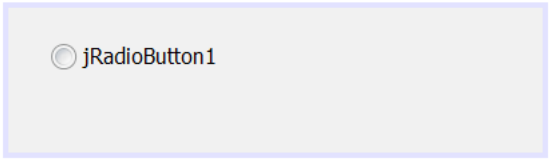
A continuación imos desenvolver o seguinte formulario:

O formulario consta de seis botóns de opción divididos en dous grupos de botóns de opción (material e cor). O primeiro grupo emprégase para elixir o material da camiseta entre varios existentes. O segundo grupo emprégase para determinar a cor da camiseta (na imaxe anterior a cor seleccionada sería a cor vermella, a que non está tachado). Ao premer sobre o botón Encargar, recolleranse os datos do formulario, validarase a súa corrección e crearase un obxecto que mapee os datos recollidos. Finalmente este obxecto lanzarase a unha base de datos para o seu proceso polo fabricante de camisetas. No caso de que ocorra algún erro durante a execución informarase ao usuario.

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

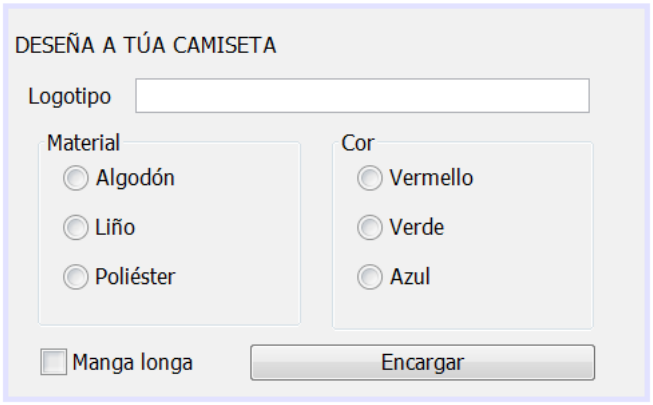


e arrastrámolo ata o formulario:

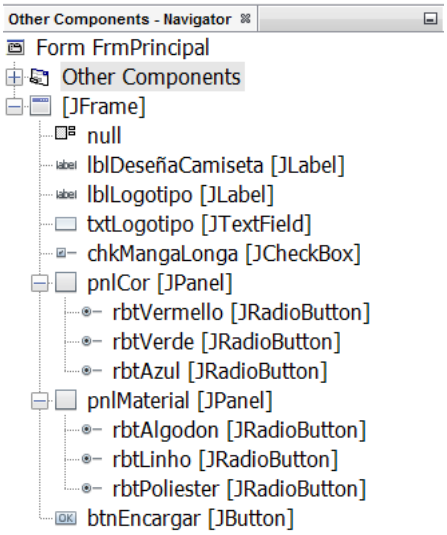


Unha vez situado dentro do formulario, e como sempre, adaptámolo para que teña o aspecto visual que desexamos que teña.

No caso que estamos desenvolvendo necesitamos engadir dous etiquetas, unha caixa de texto, un botón, dous paneis con borde titulado, unha casilla de verificación e seis botóns de opción. Unha vez engadidos e colocados no seu lugar correspondente, este será o resultado do deseño:



O noso formulario quedará configurado do seguinte xeito:



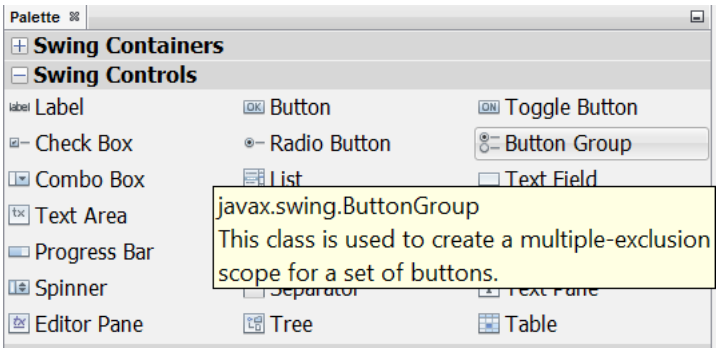
Agora deberemos modificar as propiedades dos compoñentes para que teñan o aspecto visual esperado. Centrándonos nos botóns de opción, estes son os cambios realizados:

- Para rbtAlgodon activamos a súa propiedade selected de xeito que ao iniciar a aplicación este botón de opción sexa o marcado do seu grupo.
- Para rbtVermello activamos a súa propiedade selected de xeito que ao iniciar a aplicación este botón de opción sexa o marcado do seu grupo. Á súa propiedade icon asignámoslle unha referencia á imaxe vermelloNon.png (cadrado vermello tachado) para asociar unha imaxe ao botón de opción. Á súa propiedade selectedIcon asignámoslle unha referencia á imaxe vermelloSi.png (cadrado vermello) para asociar unha imaxe ao botón de opción cando está seleccionado. Por último desactivamos a súa propiedade RollOverEnabled para evitar que se amosen imaxes que non deben amosarse ao navegar co rato sobre o compoñente.
- Para rbtVerde, á súa propiedade icon asignámoslle unha referencia á imaxe verdeoNon.png (cadrado verde tachado) para asociar unha imaxe ao botón de opción. Á súa propiedade selectedIcon asignámoslle unha referencia á imaxe verdeSi.png (cadrado verde) para asociar unha imaxe ao botón de opción cando está seleccionado. Por último desactivamos a súa propiedade RollOverEnabled para evitar que se amosen imaxes que no deben de amosarse ao navegar co rato sobre o compoñente.
- Para rbtAzul, á súa propiedade icon asignámoslle unha referencia á imaxe azulNon.png (cadrado azul tachado) para asociar unha imaxe ao botón de opción. Á súa propiedade selectedIcon asignámoslle unha referencia á imaxe azulSi.png (cadrado azul) para asociar unha imaxe ao botón de opción cando está seleccionado. Por último desactivamos a súa propiedade RollOverEnabled para evitar que se amosen imaxes que non deben de amosarse ao navegar co rato sobre o compoñente.

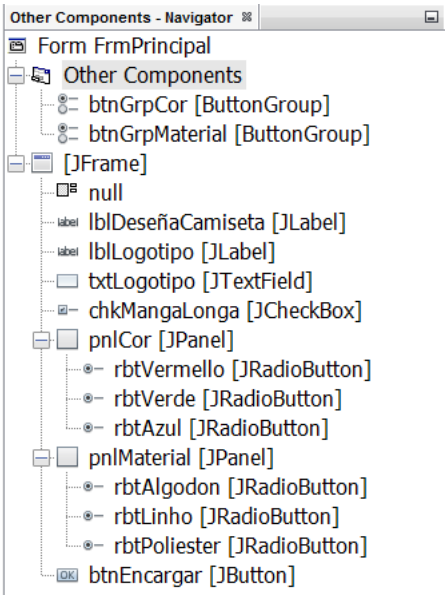
Se executamos a aplicación este será o resultado:

Como pódese observar o comportamento visual da aplicación aparentemente está resolto. Pero só aparentemente xa que se facemos clic sobre diferentes botóns de opción podemos observar que se poden seleccionar varios simultaneamente, o cal conceptualmente equivale a elixir varias opcións á vez. Isto, evidentemente é un erro:

Para solventar este erro debemos agrupar os nosos botóns de opción. Neste caso debemos de crear dous grupos, un para os botóns de opción do material e outro para os botóns de opción da cor. Para elo engadimos ao noso formulario dous compoñentes de tipo grupo de botóns (javax.swing.ButtonGroup), un por cada grupo que queremos crear. Para engadir un compoñente de tipo grupo de botóns no noso formulario primeiramente seleccionarémolo da paleta de compoñentes do entorno de programación, e arrastrámolo ata o formulario:



Visualmente non se amosa nada, pero a presencia deste compoñente reflíctese na árbore de obxectos do proxecto. No caso que estamos desenvolvendo imos engadir dous grupos de botóns. Finalmente, o noso formulario quedará configurado do seguinte xeito:



Finalmente teremos que indicar que botóns de opción pertencen a que grupo de botóns. Neste caso modificaremos a propiedade ButtonGroup dos botóns de opción rbtAlgodon, rbtLiño e rbtPoliester para que valga btnGrpMaterial. Por outra banda, modificaremos a propiedade ButtonGroup dos botóns de opción rbtVermello, rbtVerde e rbtAzul para que valga btnGrpCor. Agora o comportamento dos botóns de opción é o axeitado.

Unha vez preparada a parte visual do noso formulario imos centrarnos na súa funcionalidade. A acción desenvólvese no momento que facemos clic sobre o botón Encargar.

Centrémonos na parte do código referente á recollida da información dos botóns de opción. Unha vez validado o formulario creamos un obxecto da clase Camiseta chamado camiseta. A clase Camiseta unicamente define unha estrutura de datos para almacenar rexistros referentes ás camisetas creadas (consultar código da clase Camiseta):

```
Camiseta camiseta=new Camiseta(logotipo, chkMangaLonga.isSelected());
```

Empregando os métodos setter do obxecto camiseta asignamos os valores dos botóns de opción seleccionados aos atributos do obxecto. Imos ver como recuperamos o botón de opción seleccionado para o grupo de botóns que facen referencia ao material da camiseta (o procedemento para recuperar o botón de opción seleccionado para o grupo de botóns que facen referencia á cor da camiseta é similar). O primeiro, debemos establecer un criterio para asignar valores en función do botón de opción seleccionado. Neste caso, a asignación decidida é a seguinte: Algodón=0, liño=1 e poliéster=2 (poderíase facer outra calquera e ademais ni tan sequera sería necesario que sexa unha asignación numérica, aínda que é recomendable). Ben, partindo desta asignación, debemos comprobar para cada un dos botóns de opción do grupo se están marcados ou non mediante o emprego do seu método isSelected. No caso de que un botón de opción estea marcado, asignamos ao atributo material do obxecto camiseta un valor numérico que será o que teremos asignado no noso criterio a ese botón de opción:

```
        //Recollida de datos dos radio buttons do material
        if(rbtAlgodon.isSelected())
        {
            camiseta.setMaterial(0);
        }
        else
        {
            if(rbtLinho.isSelected())
            {
                camiseta.setMaterial(1);
            }
            else
            {
                camiseta.setMaterial(2);
            }
        }
```

Hai que asegurarse de que validamos tódolos botóns de opción. Neste caso, a mellor solución para facelo é un if aniñado que pase por tódolos botóns de opción.

Finalmente, este sería o código completo do actionPerformed do botón Encargar:

```
    private void btnEncargarActionPerformed(java.awt.event.ActionEvent evt) {
        // TODO add your handling code here:
        String logotipo=txtLogotipo.getText().trim();
        if(logotipo.compareTo("")==0)
        {
            JOptionPane.showMessageDialog(this, "Debe indicar o logotipo da camiseta");
            return;
        }

        //Nova Camiseta
        Camiseta camiseta=new Camiseta(logotipo, chkMangaLonga.isSelected());

        //Recollida de datos dos radio buttons do material
        if(rbtAlgodon.isSelected())
        {
            camiseta.setMaterial(0);
        }
        else
        {
            if(rbtLinho.isSelected())
            {
                camiseta.setMaterial(1);
            }
            else
            {
                camiseta.setMaterial(2);
            }
        }



        // Recollida de datos dos radio buttons do cor
        // empregando unha variable local
        int cor;
        if(rbtVermello.isSelected())
        {
            cor=0;
        }
        else
        {
            if(rbtVerde.isSelected())
            {
```

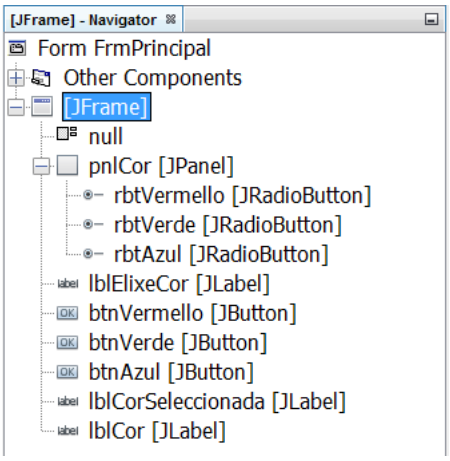
```
                cor=1;
            }
            else
            {
                cor=2;
            }
        }

        camiseta.setCor(cor);

        //Traza para comprobar a correccion do obxeto creado
        System.out.println(camiseta);

        //Insertar o obxeto creado na BD
        //Non é parte desta unidade didactica

        JOptionPane.showMessageDialog(this, "Rexistro gardado correctamente");
    }
```

Xestión de botóns de opción empregando NetBeans (segundo exemplo)

No exemplo anterior vimos como recuperar o estado dun botón de opción. Agora imos desenvolver un exemplo para explicar como asignar un estado por código a un botón de opción e como xestionar o seu evento ItemStateChanged. Para elo desenvolveremos o seguinte formulario:

No caso de que premamos sobre o botón Vermello marcarase o botón de opción Vermello. No caso de que premamos sobre o botón Verde marcarase o botón de opción Verde. No caso de que pulsemos sobre o botón Azul marcarase o botón de opción Azul. Ademais, cada vez que cambie o valor dun botón de opción, xa sexa por acción directa do usuario sobre el ou por acción indirecta a través do emprego dos botóns (Vermello, Verde e Azul) amosarase nunha etiqueta a cor seleccionada. Ademais a etiqueta terá como cor de fondo esa cor seleccionada.

O noso formulario quedará configurado do seguinte xeito:



Centrándonos no código da aplicación, imos implementar a primeira parte do programa, é dicir, a parte na que activamos ou desactivamos os botóns de opción en función dos botóns premidos.

O código asociado ao botón Vermello, o cal activa o botón de opción Vermello é o seguinte:

```
        rbtVermello.setSelected(true);
```

Como pódese observar, para activar un botón de opción unicamente hai que pasarlle o valor true a través do seu método setSelected.

Respecto á segunda parte do programa, é dicir, a parte na que se amosa a cor seleccionada na etiqueta lblCor en función do botón de opción seleccionado (xa sexa este cambio debido á acción directa do usuario sobre os botóns de opción ou por a súa acción indirecta a través do emprego dos botóns btnVermello, btnVerde e btnAzul), o xeito máis eficiente de resolvela é mediante o emprego dos eventos asociados aos botóns de opción, en concreto o evento ItemStateChanged, que é o evento que se activa no momento que cambia o estado dun botón de opción.

Para facer isto debemos implementar o método que se dispararа cando ocorre o evento ItemStateChanged de cada un dos botóns de opción.

Para o evento ItemStateChanged do rbtVermello debemos escribir as seguintes liñas de código:

```
        if(evt.getStateChange()==ItemEvent.SELECTED)
        {
            lblCor.setBackground(Color.red);
            lblCor.setText("VERMELLO");
        }
```

Como xa dixemos con anterioridade, o evento ItemStateChanged detecta se hai un cambio de estado no botón de opción. Cámbiase de estado cando selecciónase un botón de opción, pero ollo, tamén cambiase de estado cando deselecciónase (isto ocorre cando é marcado outro botón de opción). Para detectar en cal das dúas situacións estamos, empregamos a información recibida polo método que implementa o evento a través do seu parámetro evt. O obxecto evt contén información referente ao evento ocorrido. Entre esa información atópase a que indica se o botón de opción foi seleccionado ou deseleccionado. Para acceder a ela facémolo a través do método evt.getStateChange(), o cal devolveranos ItemEvent.SELECTED no caso de que o botón de opción haia sido marcado, ou ItemEvent.DESELECTED no caso de que haia sido desmarcado. No caso que estamos desenvolvendo unicamente interésanos saber

se foi marcado. No caso de que o fora, simplemente cambiamos a cor de fondo da etiqueta lblCor a vermello e establecemos o seu texto como VERMELLO. O código para os outros dous botóns de opción é similar (axustando o cambio de cor e de texto para a etiqueta lblCor).

Se executamos a aplicación este será o resultado se marcamos o botón de opción Verde, xa sexa directamente ou mediante o emprego do botón btnVerde: