



## **Ejercicio Nivel 5** **Visor mitologías** **Ayudas ejercicio**

### **Consejos prácticos para la construcción de la interfaz gráfica de la aplicación**

1. Recomendación para la distribución de los paneles en la ventana principal.



Cada cuadro en rojo es un panel y representa lo siguiente.



1. Etiqueta (JLabel) con la imagen del banner.
  2. Pánel con la información de la mitología, donde cada cuadro azul representa:
    - a. Es un pánel para el nombre y la imagen de la mitología.
    - b. Es un pánel con la información de la mitología.
    - c. Es un pánel para las acciones de navegación sobre mitologías.
  3. Pánel con la información del dios, donde cada cuadro azul representa:
    - a. Es un pánel para el nombre y la imagen del dios.
    - b. Es un pánel con la información del dios.
    - c. Es un pánel para las acciones de navegación sobre los dioses.
  4. Pánel con las acciones.
2. Si desea separar verticalmente los elementos gráficos distribuidos con un GridLayout, puede hacer uso del método setVgap(...), como se muestra a continuación:

```
GridLayout layout = new GridLayout( 2, 2 );
layout.setVgap( 5 );
panelX.setLayout( layout );
```

3. Para incluir una imagen en un panel o una ventana puede utilizar un JLabel de la siguiente manera:
- ```
JLabel lblImagen = new JLabel( );
ImageIcon icono = new ImageIcon( rutaImagen );
lblImagen.setIcon( icono );
```

Donde rutaImagen es la ruta donde se encuentra ubicada la imagen, por ejemplo el banner se encuentra en `"/data/imagenes/banner.png"`.

4. Para el manejo de distintos tipos de fuentes en JLabel se usa la clase Font de java.awt. Para crear una fuente personalizada y asignarla a un JLabel puede utilizar los siguientes pasos:

```
JLabel lblEjemplo = new JLabel( textoLabel );
Font fuenteNueva = new Font("Arial", Font.BOLD, 20);
lblEjemplo.setFont(fuenteNueva);
```

Para este caso, se asigna una fuente Arial de tamaño 20 a la etiqueta de nombre lblEjemplo. Si desea cambiar el tipo de fuente debe modificar el primer parámetro del método constructor de Font. Si desea cambiar el estilo de fuente debe modificar el segundo parámetro del método constructor de Font. Si desea cambiar el tamaño de fuente debe modificar el tercer parámetro del método constructor de Font.

Un ejemplo de los textos que se pueden obtener al aplicar formatos diferentes de letra es el siguiente:

Una etiqueta creada con los siguientes comandos:

```
JLabel lblEjemplo = new JLabel( "Texto ejemplo" );
Font fuenteNueva = new Font("Arial", Font.BOLD, 15);
lblEjemplo.setFont(fuenteNueva);
```

Se verá así:

**Texto ejemplo**



La misma etiqueta pero con formato distinto de fuente tiene los siguientes comandos:

```
JLabel lblEjemplo = new JLabel( "Texto ejemplo" );  
Font fuenteNueva = new Font("Comic Sans MS", Font.ITALIC, 30);  
lblEjemplo.setFont(fuenteNueva);
```

Se verá así:



Si desea consultar la documentación de la clase Font, para conocer los estilos y tipos de letra disponible, lo puede hacer en el siguiente enlace:

<http://docs.oracle.com/javase/6/docs/api/java/awt/Font.html>

5. Para ajustar el texto de una etiqueta a un ancho máximo, como sucede por ejemplo con la descripción de una mitología y la descripción de un dios, puede utilizar un método de la clase String que se encarga de alinear el texto dado el tamaño del ancho máximo que puede ocupar.

Un ejemplo del formato que se puede dar a los textos es el siguiente:

```
String textoFormato = String.format("<html><div align=\"justify\"  
style=\"width:%dpx;\">%s</div><html>", tamaño, textoOriginal);  
lblDescripcion.setText ( textoFormato );
```

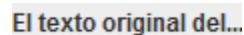
Donde textoFormato es el texto que se obtiene después de aplicar el formato deseado a un texto original, en este caso textoOriginal. El ancho máximo en píxeles que puede tener el texto ajustado se determina con la variable tamaño. Para el caso de este ejercicio se recomienda un tamaño entre 230 y 250 píxeles. Por último, se asigna el textoFormato a una etiqueta; en este ejemplo la etiqueta se llama lblDescripcion.

El siguiente ejemplo muestra la diferencia entre una etiqueta sin aplicar el formato y la misma etiqueta con el formato aplicado.

Sin aplicar el formato, la etiqueta creada con los siguientes comandos:

```
JLabel lblEjemplo = new JLabel( );  
String textoOriginal = "El texto original del ejemplo sin formato.";  
lblEjemplo.setText ( textoOriginal );
```

Se verá así:



En este caso, el tamaño asignado a la etiqueta es más pequeño que el espacio que ocupa el texto, es por esto que no es posible ver todo el texto de la etiqueta. Al asignar al texto de la etiqueta un formato que se ajusta al ancho máximo dado, que en este caso son 100 píxeles, se utilizarán los siguientes comandos:

```
JLabel lblEjemplo = new JLabel( );  
String textoOriginal = "El texto original del ejemplo sin formato.";
```



```
String textoFormato = String.format("<html><div align=\"justify\" style=\"width:%dpx;\">%s</div><html>", 100, textoOriginal);  
lblEjemplo.setText ( textoFormato );
```

Se verá así:

El texto original del  
ejemplo sin formato.

## Explicación de algunos aspectos del mundo

1. El constructor de la clase principal, VisorMitologias, se encarga de crear todas las mitologías y los dioses de cada mitología cargando la información de un archivo.
2. La clase VisorMitologia maneja una posicionMitologiaActual, que representa la posición de la mitología que se está visualizando, en el arreglo de mitologías. Los métodos de darPrimeraMitologia, darMitologiaAnterior, darMitologiaSiguiente, darUltimaMitologia se encargan de actualizar el índice de la posición de la mitología a visualizar y la retornan.
3. La clase Mitologia maneja una posicionDiosActual, que representa la posición del dios que se está visualizando, en el arreglo de dioses. Los métodos de darPrimerDios, darDiosAnterior, darDiosSiguiente, darUltimoDios se encargan de actualizar el índice de la posición del dios a visualizar y lo retornan.
4. La clase Mitologia cuenta con un atributo color de tipo Color, una clase del paquete java.awt. Esto hace que cada mitología tenga su propio color al momento de crear el mundo. Este atributo sirve para asignarle el color a la etiqueta que representa el nombre de la mitología en la interfaz gráfica.

