Practica 4. Creación de una aplicación "GUI" usando componentes disponibles por el lenguaje.

Competencia a desarrollar

Diseña e implementa componentes y librerías para lograr la reutilización de código.

Introducción

Las aplicaciones tipo GUI usan invariablemente componentes de este tipo que facilitan enormemente su desarrollo. En las prácticas anteriores se han utilizado componentes GUI de los paquetes de java: swing y awt. Ahora corresponde ejercitar el proceso de su desarrollo, partiendo desde su diseño, implementación y la manera de generar con ellos bibliotecas para cumplir la propiedad de encapsulamiento.

Aquí se propone crear un tipo de componente para un ambiente visual que además utiliza un poco manejo de gráficos con la intención de mostrar que un componente puede ser para cualquier ambiente.

Correlación con los temas y aplicación en el contexto.

Desde la primera práctica se han utilizados componentes de tipo GUI, así que los dos primeros temas de esta segunda competencia (unidad) se llegan a reforzar en esta práctica: '*definición conceptual de componentes, paquetes / bibliotecas' y 'uso de bibliotecas proporcionadas por el lenguaje'*. Además, estos temas se llegan a integrar con el último tema que se trata de lleno en esta práctica: *Creación de componentes (visuales y no visuales) definidos por el usuario.*

Los componentes utilizados y a desarrollar están dentro del contexto de la propuesta de Oracle –Java, utilizando el estándar ahí propuesto. El acceso a su funcionalidad y elementos internos de cada componente es mediante sus métodos get y set a nivel de programación.

Material y equipo necesario

Equipo de cómputo: Laptop o PC

Software: Aquí se emplea el "IDE" NetBeans (el cual cuenta con el asistente para crear un *.jar), pero puedes utilizar cualquier otro que maneje java con un entorno de diseño de GUI-Java, versión del jdk 1.8.0 .
Asegurarse que dentro de la carpeta bin de java se encuentre la herramienta: jar.exe

Metodología

Se parte de la descripción de lo que se pretende realizar, presentado los pasos a seguir para el desarrollo de componentes y una descripción sobre el caso a diseñar e implementar, se expone

un caso de un componente visual-gráfico. Luego en el desarrollo de la práctica se va dirigiendo al estudiante para llevar a cabo el proceso propuesto, haciendo que contribuya con su ingenio a ciertas partes de la funcionalidad del componente y del mismo proceso. Las partes que tiene que completar el estudiante están identificadas con el signo '?'

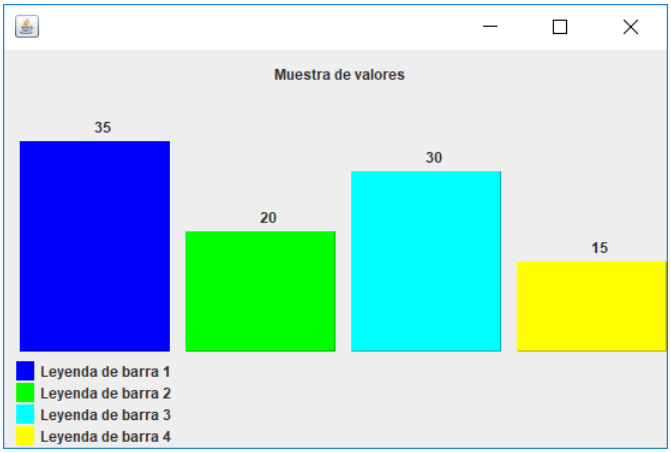
Descripción

El proceso que aquí se sigue para el desarrollo de un componente java es:

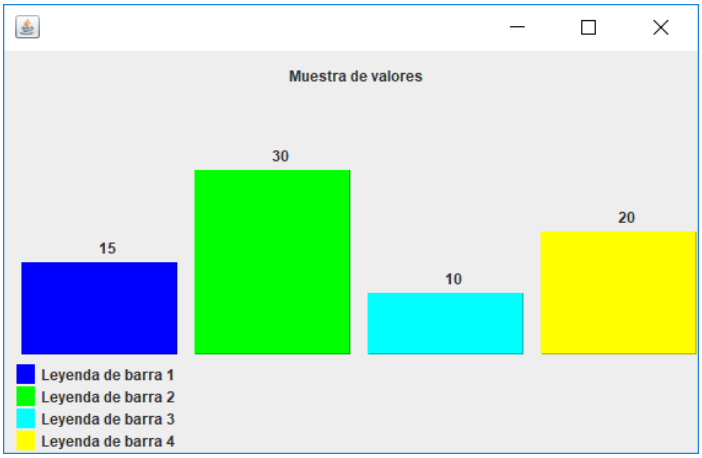
1. Descripción de la funcionalidad, donde se especifica lo que se espera que realice el componente y se hace un esbozo de su diseño dependiendo del tipo que es (GUI o solo funcional)
2. Diseño de los elementos que lo conforman. Aquí se especifican sus interfaces de entrada y salida (métodos get y set)
3. Implementación. Es la creación de la clases o clases necesarias siguiendo las convenciones propuestas por Oracle.
4. Pruebas. Esto es un punto vital para garantizar que se cumple con la funcionalidad especificada
5. Generacion del componente java *.jar e integración de este en alguna biblioteca para su utilización.

A continuación se describe el componente propuesto:

Descripción de la funcionalidad del componente tipo GUI denominado "***Grafico***"".
Funcionalidad: Dibujar un gráfico de barras de colores que indiquen en su dimensión los valores que se le proporcionen y las leyendas que las identifiquen. El número de barras será de acuerdo a las leyendas de identificación que se proporcionen y cada vez que se cambie algunos de los valores estas deben reflejar en su dimensión el valor correspondiente. Abajo se muestra un ejemplo de cómo debe ser.



En este ejemplo hay cuatro elementos con valores distintos que se muestran centrados en la parte superior de cada barra. Las leyendas que identifican a cada barra están en la parte inferior alineadas a la izquierda con su color y descripción correspondiente. También hay un título para describir al gráfico centrado en la parte superior.
Cada vez que cambien los valores de las barrar, éstas deben redibujarse, por ejemplo si la primera barra vale ahora 10, la segunda 30, la tercera 10 y la última 20 unidades el nuevo gráfico se deberá ver así:



Lo anterior ilustra el hecho de que la variación de los valores de cada barra y su redibujo es una de las funciones más utilizadas.

Diseño del componente “Grafico”

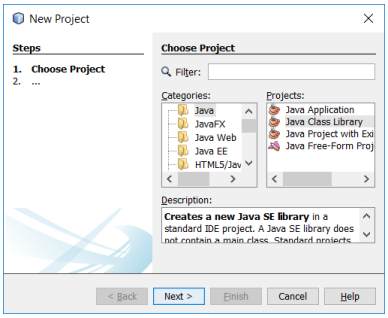
Elemento	Descripción	Característica	Implementación
Titulo	Título de identificación	Debe ser proporcionado una vez. Debe estar en la parte superior y con alienación centrada	JLabel
Barra	Es un gráfico tipo rectángulo	Cada barra debe tener un color diferente generado por el mismo componente (no por el usuario). El número de barras es variable, aquí se consideran que puede variar de 1 a 10 elementos.	Generado por un método de una instancia de ña clase “Graphics”
Valor de cada barra	Muestra el valor numérico de cada barra	Debe estar en la parte superior de cada barra y con alienación centrada	JLabel
Leyendas de cada barra	Identifican con el color correspondiente y su texto	Debe verse cada color de la barra y su texto enseguida de éste en la parte inferior izquierda, ajustándose de acuerdo al número de leyendas.	Un JLabel para el color y otro para el texto

Interface	Función	Parámetros
setTitulo	Establece el título a desplegar	Una cadena de texto
setEtiquetas	Establece los textos desplegar de cada barra y el número de estas fija el número de barras a desplegar	Un conjunto de cadenas de texto, de una a diez cadenas.
setValores	Establece el valor correspondiente a cada barra que determina su altura y el número a desplegar arriba de éstas.	Un conjunto de números enteros >=0, de la misma dimensión que el número de leyendas
getValores	Regresa los valores actuales de cada barra	-

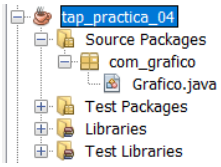
Desarrollo

Implementación del componente “Grafico”

1. Dentro del ide NetBeans, crea un proyecto Java tipo biblioteca(Library)



Da clic en el botón Next y nombralo como tap_practica_04, en él crea un paquete fuente con el nombre ‘com_grafico’ y dentro de este último crea una clase, llámala Grafico



a) Hereda la clase Grafico de JComponent, esto es para cumplir con la función principal sobre producir un gráfico de barras.

b) Implementa la interface Serializable, ¿para qué tiene que hacerse esto? , anota en tu bitácora la respuesta

2. Declaración de los atributos, crea los atributos de la clase.

```
private int nfiguras; // numero de barras
private JLabel [] etis; // etiquetas para el valor a mostrar
private int [] vals={0,0,0};//valores de cada barra, se dan valores por omision
//Se consideran diez colores para diez valores
private static Color color[]={Color.BLUE,Color.GREEN,Color.CYAN,
  Color.YELLOW,Color.PINK,Color.WHITE,Color.RED,Color.ORANGE,Color.MAGENTA};
private static JLabel  leyendaBarra[]; //muestra el titulo de la leyenda
private static JLabel  colorB[]; // identificación del color de la barras
private JLabel titulo; // titulo
private String tTitulo; // Texto del titulo
private String tLeyenda[];// titulos de las leyendas que idefican a cada barra
```

3. Creación de constructores

a) Crea un constructor sin parámetros con atributos con valores por omisión. ¿Por qué es necesario un constructor sin parámetros ? (anota tu respuesta en tu bitácora)

```
//Valores por omision
 titulo = new JLabel("");
 etis   = new JLabel[0];
 leyendaBarra   = new JLabel[0];
 colorB = new JLabel[0];
 tLeyenda = new String[0];
```

b) Crea un constructor que pase como valores el título del gráfico y leyendas, y tenga valores por omisión.

```
public Grafico(String encabezado,String tLeyenda[])
{
  nfiguras=tLeyenda.length;
  this.tLeyenda = tLeyenda; // texto de cada leyenda
  leyendaBarra = new JLabel[tLeyenda.length];
  tTitulo= encabezado;
  setLeyendas(this.tLeyenda);
  iniciarElementos(); // método para iniciar los valores

}
```

c) Método iniciarElementos

```
public void iniciarElementos()
{ // Creacion e inicio de los valores de cada barra
  vals = new int[nfiguras];
  for(int nv=0;nv<nfiguras;nv++)
  { //inicia los valores
     vals[nv]= 0;
  }
 // crea y agrega la etiqueta del titulo
   titulo = new JLabel(tTitulo);
   add(titulo);
 //
// crea  e inicia los demas arreglos
  etis = new JLabel[nfiguras]; //
  colorB = new JLabel[nfiguras];
  for(int i=0;i<nfiguras;i++)
  {
   etis[i] = new JLabel(""+0);
   add(etis[i]);
  }
  // identificacion del color de cada barra colorB[ ]
  //? Te corresponde completar este código
}
```

d) ? . Crea el código indicado, que sirva para asignar color a las etiquetas que acompañan a las leyendas de las barras, cuyo identificador es ¨colorB´ y agregarlas al componente.

4. Creación de los métodos para asignar título, leyendas y valores

a) Método para asignar titulo

```
public void setTitulo(String encabezado)
{
  tTitulo= encabezado;
}
```

b) Método para asignar leyendas

```
public void setLeyendas(String tLeyenda[])
{
  nfiguras=tLeyenda.length;
  this.tLeyenda = tLeyenda;
  leyendaBarra = new JLabel[tLeyenda.length];
  for(int e=0; e< tLeyenda.length;e++)
  {
    leyendaBarra[e]= new JLabel(tLeyenda[e]);
  }
  iniciarElementos();
}
```

c) ?. Crea el método: public void setValores(int valores[])

{//Actualiza los valores y los asigna a las etiquetas

……

}

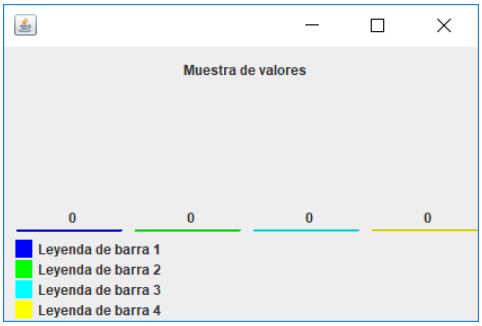
5. Creación del método para crear el gráfico:

```
@Override
public void paintComponent(Graphics f)
{

   int i;
   int j=1;
   int ancho =  getWidth();
   int alto  =   getHeight();
   titulo.setBounds((int)((getWidth()-f.getFontMetrics().stringWidth(titulo.getText()))/2),10,titulo.getText().length()*8,20);
   int separa =(int) (getWidth()/(nfiguras)*0.10);
   int anchoB = (int)(getWidth() / (nfiguras) - separa);

   // Ubica a cada elemento en su posición
   for(i=0; i<nfiguras; i++)
   {    f.setColor(color[i]); // Establce el color de cada barra
       etis[i].setBounds(separa+i*(anchoB+separa)+(anchoB/2),(alto-20*color0.length)-vals[i]*5-etis[i].getHeight()-2,30,20);
       f.fill3DRect(separa+i*(anchoB+separa),((alto-20*color0.length)-vals[i]*5),anchoB,vals[i]*5,true);
       // ? TE corresponde fijar:
       // las posicones de los colores de idenficacion color0[?].setBounds(?)
       //y las posiciones de  los textos de leyendas de las barras leyendaBarra[nfiguras-1-i].setBounds(
       j++;
   }
}
}
```

? Crea el código para fijar la posición de los elementos que muestran el color de identificación y texto de las leyendas correspondientes a cada barra(lo cual está indicado como comentarios en el código de arriba).

6. Pruebas de la clase

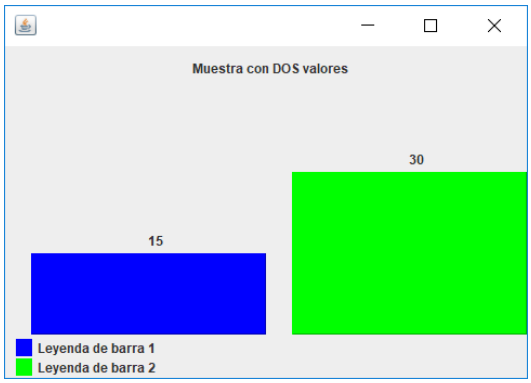
a) Crea una clase prueba que cuente con un JFrame que agregue a su contenedor un objeto tipo '**Grafico'** y se le comuniquen valores del título y etiquetas sin valores. La gráfica debe mostrase como la siguiente:

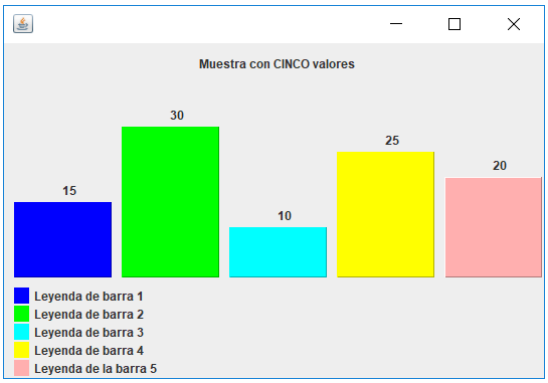


b) Cambio de valores. Usa el método setValores() para asignarle valores a las barras, usa los valores mostrados en la descripción del componente para que al ejecutar de nuevo el programa se vea como en dicha muestra.

c) Comprobar que se distribuyen en forma equidistante y sus leyendas debajo del gráfico. Varia el ancho de la ventana de prueba para comprobar que se distribuyen en forma equidistante

d) Prueba de numero de barras, crea un gráfico con dos etiquetas con sus valores de barras y otro de cinco, deben verse como:





7. Generación del componente.

a) Creación del *.jar usando el asistente del ide NetBeans

i) Ubicar el cursor donde está el proyecto, cuidando que solo esté la clase Grafico

ii) Dar clic derecho, en la ventana emergente seleccionar ‘Build’

iii) La acción anterior y su resultado se refleja en la ventana de salida. En la última línea se muestra la carpeta donde se generó el archivo jar, el cual lleva el nombre del proyecto

```
ant -f C:\\Users\\Usuario\\Documents\\NetBeansProjects\\tap_practica_04 -Dnb.internal.action.name=build jar
init:
Deleting: C:\Users\Usuario\Documents\NetBeansProjects\tap_practica_04\build\built-jar.properties
deps-jar:
Updating property file: C:\Users\Usuario\Documents\NetBeansProjects\tap_practica_04\build\built-jar.properties
compile:
Copying 1 file to C:\Users\Usuario\Documents\NetBeansProjects\tap_practica_04\build
Copy libraries to C:\Users\Usuario\Documents\NetBeansProjects\tap_practica_04\dist\lib.
To run this application from the command line without Ant, try:
java -jar "C:\Users\Usuario\Documents\NetBeansProjects\tap_practica_04\dist\tap_practica_04.jar"
jar:
BUILD SUCCESSFUL (total time: 0 seconds)
```

b) Opción a través de la línea de ordenes de la consola o Símbolo del sistema

i) Acceder a la ventana del sistema (Símbolo de sistema)

ii) Ubicarse en el directorio donde se encuentra el archivo Grafico.class (generado al hacer la compilación)

cd C:\Users\Usuario\Documents\NetBeansProjects\tap_practica_04\build\classes\com_grafico

iii) Escribe la orden para generar el jar :

jar cvf com_grafico.jar Grafico.class

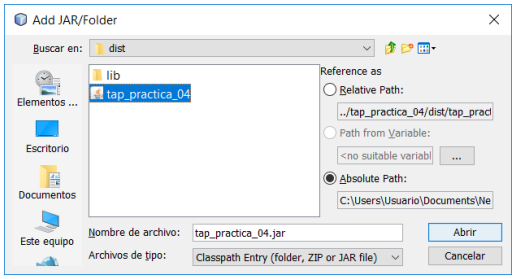
Esto se muestra en la ventana del 'Símbolo del sistema'

```
C:\Users\Usuario\Documents\NetBeansProjects\tap_practica_04\build\classes\com_grafico>dir
 El volumen de la unidad C no tiene etiqueta.
 El número de serie del volumen es: B64A-1BA1

 Directorio de C:\Users\Usuario\Documents\NetBeansProjects\tap_practica_04\build\classes\com_grafico

03/12/2018  02:34 p. m.    <DIR>          .
03/12/2018  02:34 p. m.    <DIR>          ..
03/12/2018  02:34 p. m.             4,026 Grafico.class
               1 archivos          4,026 bytes
               2 dirs  866,432,233,472 bytes libres

C:\Users\Usuario\Documents\NetBeansProjects\tap_practica_04\build\classes\com_grafico>jar cvf com_grafico.jar Grafico.class
manifiesto agregado
agregando: Grafico.class(entrada = 4026) (salida = 2237)(desinflado 44%)
```

8. Prueba del componte

a) En otro proyecto crea una clase de prueba semejante a la creada para probar la clase del componente

b) En el proyecto creado, ubica el cursor en la carpeta 'Biblioteca' o Libraries, da clic derecho y selecciona agregar *.jar. En la ventana mostrada selecciona el trayecto donde está el *.jar, y da clic en el botón 'Abrir', como se muestra a continuación



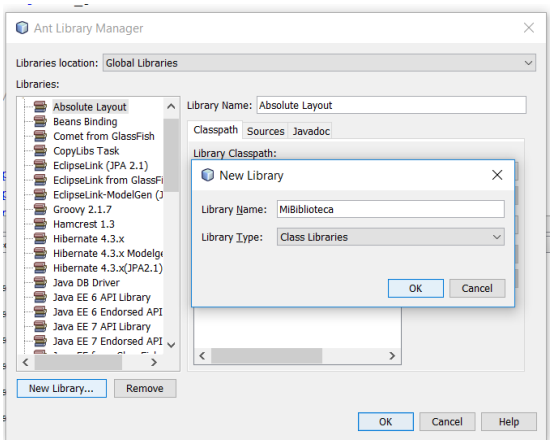
c) Por ultimo en la clase de prueba agrega el 'import' correspondiente

d) Realiza las mismas pruebas que las realizadas en el punto 6.

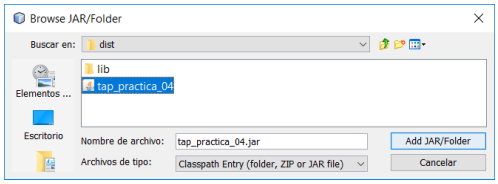
9. Creación como una biblioteca compartida dentro del 'IDE' de NetBeans

Aunque una biblioteca (Library) tiene como propósito ser una colección de componentes, sobre todo a nivel de un IDE, en esta práctica se hará solo con un componente pero posteriormente puedes agregarle más elementos.

a) Dentro del menú principal del IDE de NetBeans, elige la opción Herramientas(Tools), dentro de ahí activa Biblioteca (Library) y da clic en el botón Nueva biblioteca (New Library), puedes llamarle MiBiblioteca como se muestra a continuación:

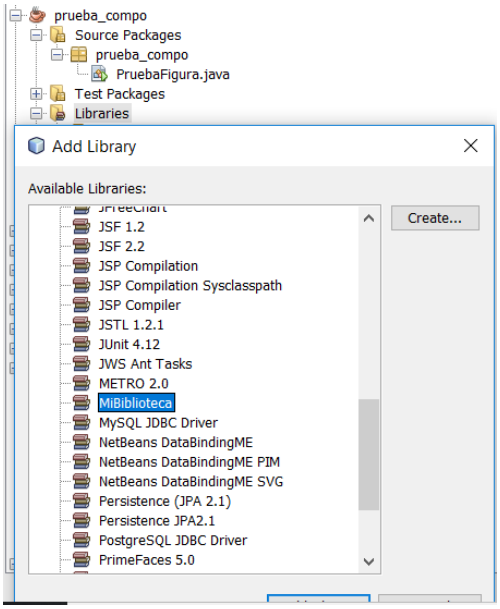


b) Agregarle componentes a la biblioteca creada. En la ventana mostrada, da clic en el botón AddJar/Folder. Ubica la ruta donde se encuentra el componente como se muestra enseguida



c) Da clic en el botón AddJar/Folder y después darle OK y listo ya se cuenta con una biblioteca

d) Ahora cada vez que requieras usarla en cualquier proyecto de aplicación la tienes disponible a nivel del IDE de NetBeans, puedes hacerlo en el proyecto prueba o en cualquiera otro



10. ? Te corresponde a ti crear una pequeña aplicación donde se requiera usar un gráfico de barras , para lo cual se tiene que importar en la carpeta de 'Libraries' del proyecto la biblioteca 'MiBiblioteca'

Sugerencias didácticas

Trabaja por equipos que pueden ser de hazta cuatro integrantes
Utiliza una bitácora de seguimiento
Documenta tu código.
Participa en un foro del grupo donde se intercambien experiencias sobre lo realizado
Esta práctica se puede dividir a criterio del docente en tantas partes como estime conveniente.

Reporte del estudiante

Bitácora del seguimiento que incluya las respuestas a lo solicitado durante el desarrollo de la práctica.
Código fuente del componente con comentarios de lo realizado por el estudiante o estudiantes (si fue realizado por equipo)
Código fuente de la aplicación de prueba que utilice al "Grafico' como componente e imágenes capturadas de las ejecuciones de cada prueba.
Entrega del elemento *.jar
Código fuente de la aplicación donde se use al componente 'Grafico' importado desde la biblioteca creada , con imágenes de capturada de la ejecución.

Bibliografía

1. Wu Thomas, Programación en Java, Mc. Graw-Hill, 2008
2. Deitel y Deitel. Como programar en Java. Prentice Hall.Septima Edición, 2008
3. Oracle Documentación, The Java™ Tutorials. https://docs.oracle.com/javase/tutorial/uiswing/components/index.html. Último acceso 16/11/18
4. https://docs.oracle.com/javase/tutorial/deployment/jar/build.html . Último acceso en 21/11/2108