Eliecer Betancourt Rivera:

DOCUMENTACION DEL PROCESO

- # Completar diagramas vectoriales.

Origen del error.

Cuando el proyecto **Patrones2** se ejecuta, establece una instancia a determinadas clases por medio del archivo **plugins.txt.**

```
List<PaintableFactory> paintableFactoryList = //
PluginsReader.fsRead(ClassLoader.getSystemResourceAsStream("plugins.txt"));
```

CLASES REFERENCIADAS:

```
tools.happy.HappyPaintableFactory
tools.normal.NormalPaintableFactory
tools.sad.SadPaintableFactory
```

Cada una de estas clases se encarga de retornar un objeto de la clase **DrawnFace.java** para la cual se envían los parámetros correspondientes para dibujar el vector que se indique.

Error:

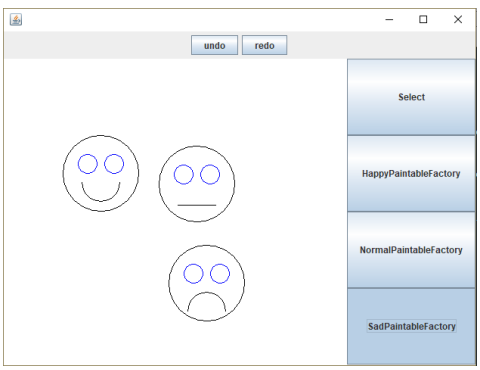
El error parte de que en la clase **SadPaintableFactory.class** está enviando parámetros para dibujar un vector diferente del que debería hacer.

```
public Paintable create(int x1, int y1, int x2, int y2) {
    if (useImage) {
   return new SadImageFace(x1, y1, x2, y2);
     } else {
       return new DrawnFace(x1, y1, x2, y2,SmileConstants.SMILE_OK);
```

Se cambia el parámetro para que dibuje el vector correcto.

```
public Paintable create(int x1, int y1, int x2, int y2) {
    if (useImage) {
   return new SadImageFace(x1, y1, x2, y2);
     } else {
       return new DrawnFace(x1, y1, x2, y2,SmileConstants.SMILE_DW);
```

Evidencia.



- # Puesta a punto patrones 1

Origen del error.

El error básicamente estaba relacionado en lo siguiente:

- Cuando se lanza la aplicación Patrones se ejecuta la clase principal **FrmMain.java.**
- En esta clase se crea un objeto que instancia a la clase **Canvas.java .**
- En el constructor de dicha clase se crea un objeto de la clase **PaintableFactory.java**
- Es en esta donde existe una variable tipo **boolean** que por defecto esta en **true**

  ```
  private boolean useImage
  ```

  Esta variable obliga a que el compilador, retorne un objeto de la clase **ImageFace.java** los valores de los parámetros que le envía, no corresponden en la búsqueda de un archivo y es en este punto donde genera la excepción.

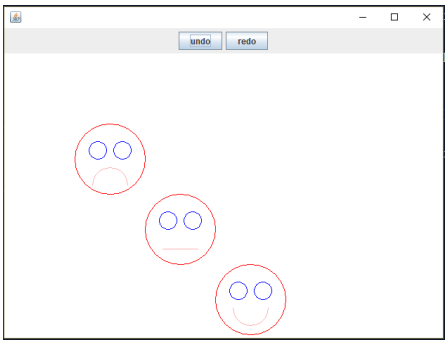
Solución

Para dar solución, solamente se cambia el valor de la variable a **true** de la siguiente forma `private boolean useImage = false;` obligando a que el compilador ingrese en la segunda opción.

```
public Paintable create(int x1, int y1, int x2, int y2, int state) {
  if (useImage) {
    return new ImageFace(x1, y1, x2, y2, state);
  } else {
    return new DrawnFace(x1, y1, x2, y2, state);
  }
```

Para retorna un objeto de la clase **DrawnFace.java**, teniendo en cuenta que las coordenadas enviadas en dicha clase dibujan los vectores correspondientes.

Evidencia



- # Integrar proyecto 2 en 3

Proceso

Se identifica la clase principal en el proyecto 3 que se llama **FrmMain.class** en esta clase se dibujan los respectivos objetos de la interfaz (botones) al momento de ejecutarse.

```
public FrmMain() {
  setLayout(new BorderLayout());

  client = new Canvas();
  add(client, BorderLayout.CENTER);

  add(initToolBarPanel(), BorderLayout.NORTH);
  add(initToolBarPanel2(), BorderLayout.EAST);

  setDefaultCloseOperation(EXIT_ON_CLOSE);
  setSize(640, 480);
  setVisible(true);
}
```

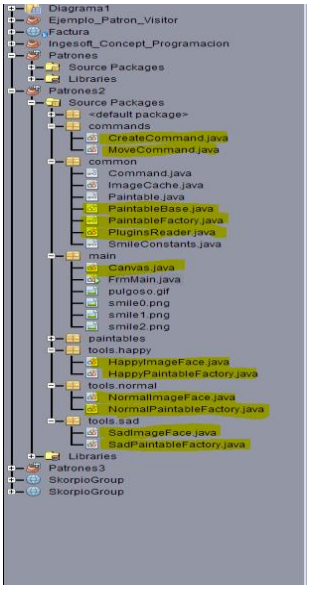
Para integrar funcionalidades se toma parte de este código del proyecto2 y se integra al proyecto 3 todo esto en la clase principal **FrmMain.class**

Fragmento del código que dibuja los botones.

```
 List<PaintableFactory> paintableFactoryList = //
 PluginsReader.fsRead(ClassLoader.getSystemResourceAsStream("plugins.txt"));

 for (final PaintableFactory paintableFactory : paintableFactoryList) {
   JToggleButton btnTool = //
   new JToggleButton(paintableFactory.getClass().getSimpleName());
   btnTool.addActionListener(new ActionListener() {
     public void actionPerformed(ActionEvent e) {
       client.setPaintableFactory(paintableFactory);
     }
   });
   ret.add(btnTool);
   buttonGroup.add(btnTool);
 }

 btnSelect.setSelected(true);

 return ret;
}

// ------------------------------------------------------------------------------

protected void btnUndoClicked() {
  client.undo();
}
```

No solamente hay que hacer esto, sino que también se debe integrar algunas clases propias del proyecto2, que se encargan de dibujar los vectores, las clases involucradas son:

CreateCommand.java
MoveCommand.java
PaintableBase.java
PaintableFactory.java
PluginsReader.java
Canvas.java



Los paquetes indicados
tools.happy
tools.normal
tools.sad

Algunas clases tienen el mismo nombre en ambos proyectos, pero difieren en cuanto a contenido y lógica, para lo cual, en la integración se copiaron las funcionalidades teniendo cuidado de no entrar en conflicto.

Evidencia.