SEED

"Simuladores para Estudio de Estructuras de Datos"

Manual de Usuario

Simulador ListaSimple<T>

Versión: 1.0

Universidad Francisco de Paula Santander
Programa Ingeniería de Sistemas
2014

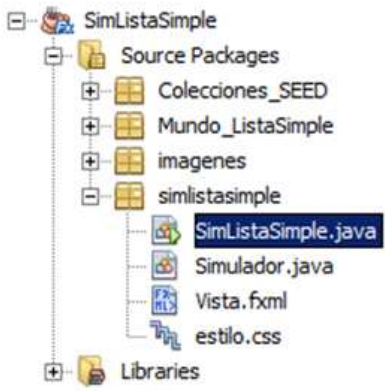
MANUAL DE USUARIO: Simulador “ListaS<T>”

Descripción General

El presente Manual de usuario pretende describir en detalle el conjunto de funcionalidades de la Aplicación desarrollada para la Simulación del comportamiento de la Estructura de Datos ListaS<T>.

Dentro de esta aplicación encontrará el estudiante un conjunto de operaciones relacionadas con las funciones básicas implementadas para la estructura Lista Simple: Insertar, Eliminar, Buscar y Editar datos dentro de cada una de las posiciones de la misma, además de crear un iterador para realizar de manera simple el recorrido.

Adicionalmente el estudiante podrá conocer de la Lista Simple, el tamaño y los datos almacenados dentro de la estructura. Para la implementación de este Simulador se ha determinado la siguiente distribución de paquetes, ya conocida por el Estudiante, de forma que sea fácilmente apropiable a futuras modificaciones con el fin de hacer buen uso de esta aplicación.



“Directorio del Simulador para ListaS<T>”

A continuación se presenta la interface principal del simulador para “Lista Simple”. El simulador para ListaS<T> permite al Estudiante crear un Lista Simple con solo correr la aplicación SimListaSimple, aunque esta estructura es dinámica “*El tamaño de la Lista Simple es limitado por cuestiones de simulación a un valor mayor a cero (0) y menor a quince (15) posiciones”.*

"Interface principal del Simulador para ListaS<T>"

Descripción de las Funcionalidades del Simulador

1. **Insertar Datos:**

   Se debe ingresar el valor del dato a insertar en la Lista Simple, el cual no puede ser menor a **-99** ni mayor a **999**, rango seleccionado por cuestiones de que no se desborde el número del nodo gráfico. Una vez insertado el dato, este será mostrado a continuación en la Lista Simple.

"Lista Simple después de insertar los datos: 32,44,33,22,55, 88, 98, 78, 67, 86, 48, 65, 999 y 23".

2. **Eliminar Datos:**

   Para Eliminar un Dato de la Lista Simple el estudiante deberá ingresar a la caja de texto el valor del dato que desea eliminar, el cual debe cumplir las mismas características de los valores insertados. Una vez eliminado el dato de la Lista Simple, este no será pintado.

"Lista Simple después de eliminar el dato: 999".

3. **Editar posición de la Lista Simple:**

   Para editar una posición de la Lista Simple el estudiante deberá insertar una **posición** valida (en la cual exista un dato) en donde se realizara la edición, y adicionalmente a esto deberá ingresar el **dato** con el cual desea remplazar el valor encontrado en dicha posición. La Lista Simple será pintada a continuación con la posición editada.

"Lista Simple después de editar la posición '8' con el dato: "11""

4. **Conocer el Tamaño de la Lista Simple:**
   Para conocer el tamaño de la Lista Simple, el estudiante podrá oprimir el botón "**Dar Tamaño"** dependiendo del valor de la Lista Simple que desee conocer en su momento.

"Tamaño de la Lista Simple determinado"

5. Reiniciar La Lista Simple:

Para reiniciar la Lista Simple, el estudiante deberá dar clic en el botón "**Reiniciar"**, esta acción elimina todos los datos de la Lista Simple dejándola vacía para que el estudiante comience a ilustrar de nuevo las funciones básicas de la estructura.

"Reinicio de la Lista Simple".

6. Adicionar nuevas funcionalidades:

Adicionalmente a las funciones incorporadas para el Simulador de Lista Simple, existe la posibilidad de que el estudiante pueda **"adicionar nuevas funcionalidades"** a la aplicación, de acuerdo a las actividades asignadas por los docentes o el interés propio de generar nuevos algoritmos en cada estructura y poder simularlos gracias a la herramienta grafica del Simulador.

A continuación se presentan los pasos que deberá seguir el estudiante para crear una nueva funcionalidad dentro del Simulador de Lista Simple:

6.1. El estudiante debe generar el nuevo Algoritmo dentro de la Estructura de Datos **ListaS**, presente en el paquete **SEED_Colecciones** y que desea adicionar a la funcionalidad del Simulador. (Para el ejemplo, se creará un algoritmo que permita "**invertir"** los datos de la Lista):

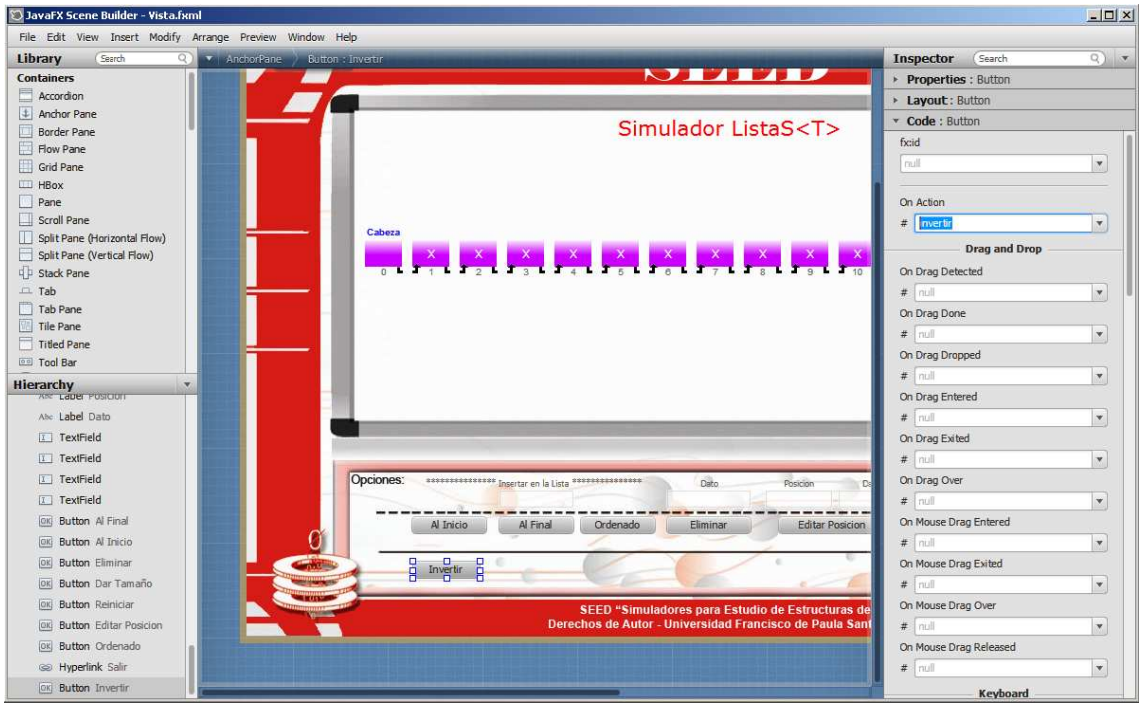
```
public void invertir(){
    for( int i=0; i<this.getTamanio(); i++){
        T obj = this.eliminar(i);
        this.insertarAlInicio(obj);
    }
}
```

6.2. A continuación el Estudiante debe generar un Método en la clase **SimuladorListaS** del paquete **Mundo_ListaSimple**, que realice el **llamado** al Método con el nuevo algoritmo creado en la Estructura de Datos **ListaS**. Para el llamado debe utilizar el objeto creado en el Mundo **miLista**.

```
public void invertirLista(){
    this.miLista.invertir();
}
```

6.3. Por último, se deberá crear el componente grafico (para el ejemplo **Button**) que permita realizar el llamado al Método creado en **SimuladorListaS.** Existe dos posibilidades para ello: Utilizar la herramienta **"JavaFX SceneBuilder"** para insertarlo, o agregar el código del Button en el Archivo **Vista.fxml**.

6.3.1. Utilizando JavaFX SceneBuilder



6.3.2. Insertando directamente el elemento en Vista.fxml

```
<Button layoutX="175.0" layoutY="610.0" mnemonicParsing="false" onAction="#invertir" prefWidth="72.0" text="Invertir" />
```

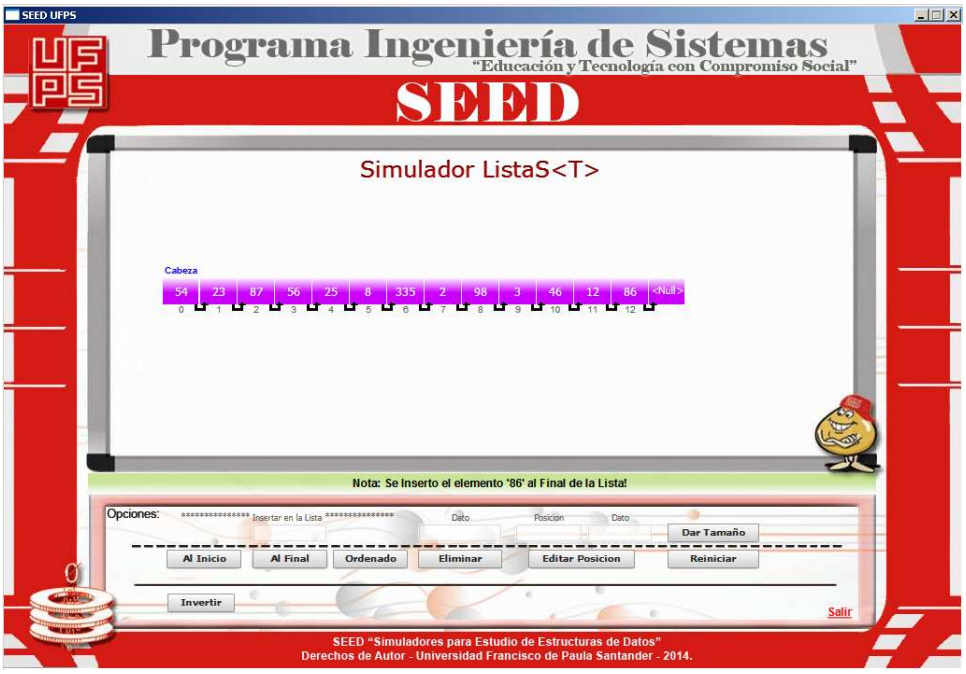
Es importante resaltar, para ambos casos, que se debe asignar el evento **“OnAction”** del Button, para el ejemplo **“invertir”**, el cual será el nombre del **Método** dentro de la clase **Controlador** que permite realizar la nueva funcionalidad del Simulador.

A continuación el Método dentro del paquete **simlistasimple** en la clase **Simulador** que permite realizar el llamado a la nueva funcionalidad.

```
@FXML
private void invertir(){
    this.simulador.invertirLista();
    this.pintarTDA();
    this.impNota("La lista ha sido Invertida!", 0);
}
```

Además del llamado al Método creado en **SimuladorListaS**, el estudiante deberá invocar el método que le permita volver pintar la Lista, el cual siempre será **“pintarTDA()”**. Opcionalmente se recomienda enviar una mensaje con la respuesta a la operación realizada utilizando **“impNota(“ Mensaje a enviar “ , tipo)”** donde tipo es cero **(0)** si en un mensaje Exitoso y uno **(1)** en caso de ser un mensaje erróneo.

A continuación se comprueba el funcionamiento del Algoritmo realizado:



“Lista original con 13 datos”.

Después de ejecutar el nuevo algoritmo la Lista es invertida y pintada con el cambio de sus datos.

"Lista invertida con 13 datos".