SEED

"Simuladores para Estudio de Estructuras de Datos"

Manual de Usuario

Simulador TablaHash<Clave,T>

Versión: 1.0

Universidad Francisco de Paula Santander
Programa Ingeniería de Sistemas
2014

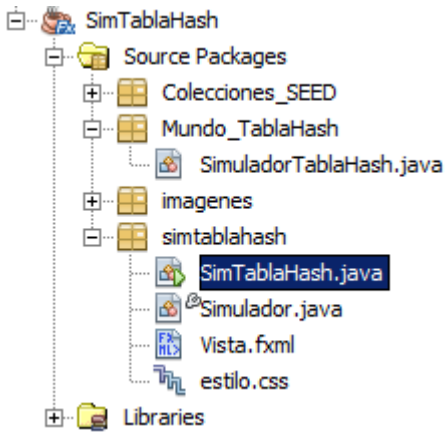
MANUAL DE USUARIO: Simulador “TablaHash<T>”

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 TablaHash<T>.

Dentro de esta aplicación encontrará el estudiante un conjunto de operaciones relacionadas con las funciones básicas implementadas para la estructura Tabla Hash: Insertar, Eliminar y Buscar un dato dentro de cada una de las ramas del Árbol. Adicionalmente el estudiante podrá conocer algunas de las propiedades del Árbol como: Altura, peso y cantidad de Hojas presentes (ilustradas gráficamente).

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 TablaHash<T>”

La implementación de este Simulador se desarrolla basada en la idea de registrar estudiantes, esto con el fin de ilustrar el comportamiento de la estructura con un ejemplo que para el estudiante se sienta identificado en el uso dela estructura y aumente el interés por el estudio del funcionamiento de la misma.

A continuación se presenta la interface principal del simulador para “Tabla Hash”.

"Interface principal del Simulador para TablaHash<T>"

Descripción de las Funcionalidades del Simulador

El simulador para TablaHash<T> permite al Estudiante:

1. **Crear un Tabla Hash:**
   Para Crear un nuevo Tabla Hash el estudiante deberá dar clic en el Botón "Crear", ingresando previamente en la caja de texto el **Número de Slots** (la cantidad de posiciones) que desea tener en la estructura, debido a que el tipo de Tabla Hash que se implemento es Estática *"El tamaño de la Tabla Hash viene limitado a un valor mayor a cero (0) y menor a treinta y siete (37) posiciones), por cuestiones de pintar la estructura".*

"Creación de un Tabla Hash de 11 posiciones".

2. Insertar Datos:

Para Insertar datos el estudiante simplemente deberá ingresar a la caja de texto el dato que desea insertar en el Árbol, este dato debe poseer cuatro (4) número que son los correspondientes al código del estudiante para este caso.

"Inserción de Estudiantes en la Tabla Hash".

3. **Eliminar Datos:**

   Para eliminar datos el estudiante deberá ingresar simplemente el dato que desea eliminar del Tabla Hash (previamente insertado) en la caja de texto. Una vez eliminado el dato del Tabla Hash este no será pintado.

"Eliminación del dato 151 de la Tabla Hash".

4. **Búsqueda de un dato (Ubicar):**

   Para ubicar un dato dentro del Tabla Hash el estudiante deberá ingresar en la caja de texto el dato que desea ubicar. La búsqueda del dato se realiza de acuerdo a las propiedades de la estructura y el costo algorítmico, para esta estructura se indica el camino de la búsqueda del dato teniendo como referencia el slot donde se encuentra en dato en la Tabla.

"Búsqueda del dato '159' dentro del Tabla Hash"

5. Determinar la cantidad de datos en la Tabla Hash

Para determinar la cantidad de datos de la Tabla Hash el estudiante deberá dar clic en el botón **Cant. Datos**, inmediatamente la aplicación indicará la cantidad de datos presentes en la Tabla Hash en el momento determinado.

"Determinar la cantidad de datos en la Tabla Hash: 6 Objetos"

6. **Determinar la cantidad de Slots (posiciones) de la Tabla Hash**

   Para determinar la cantidad de slots (posiciones) de la Tabla Hash el estudiante deberá dar clic en el botón **Num. Slots**, inmediatamente la aplicación indicará el número de Slots (posiciones) que posee la Tabla Hash.

"Determinar la cantidad de slots (posiciones) de la Tabla Hash: 11 Slots"

7. **Adicionar nuevas funcionalidades:**

   Adicionalmente a las funciones incorporadas para el Simulador de TablaHash, 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 TablaHash:

   7.1. El estudiante debe generar el nuevo Algoritmo dentro de la Estructura de Datos **TablaHash**, presente en el paquete **SEED_Colecciones** y que desea adicionar a la funcionalidad del Simulador. (Para el ejemplo, se creará un algoritmo que permita **crear una Tabla Hash** a partir de la inserción automática de cierta

cantidad de datos. Para ello se utiliza el Método ya existente **insertar**):

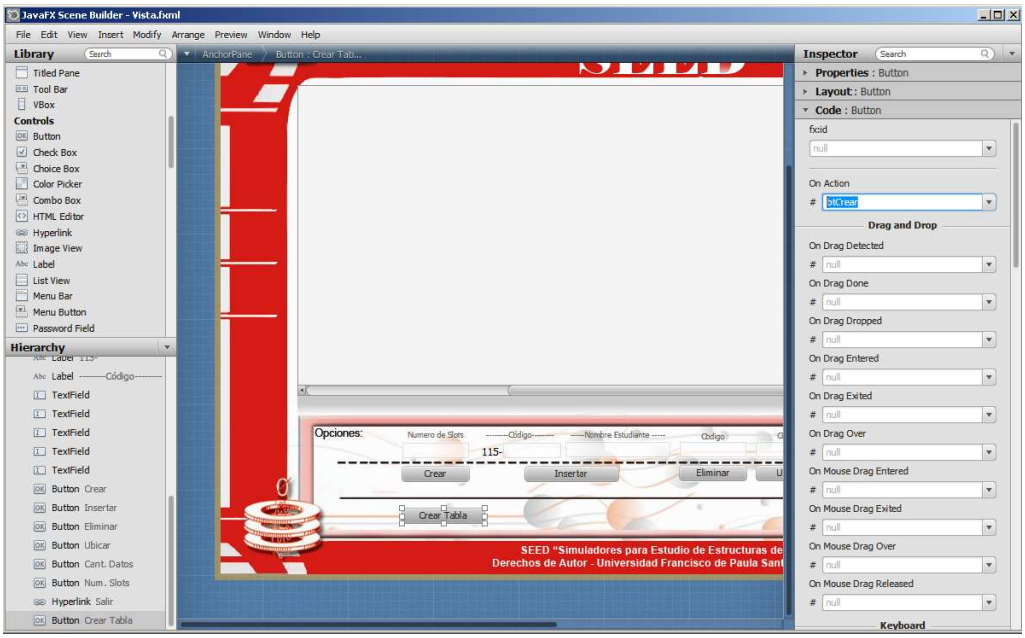
```
public T insertar( Clave clave, T objeto ) {
    int indice=0;
    InformacionDeEntrada<Clave,T> objetoAnterior=null;
    if(clave==null){
        throw new RuntimeException("La Clave de Objeto no puede ser vacia!!!");
    }
    else{
        indice =index(clave);
        objetoAnterior = this.registrarEntrada(indice, clave );
        if( objetoAnterior== null ){ // Si la clave del objeto no se encuentra en la tabla lo insertamos
            InformacionDeEntrada<Clave,T> nuevoObjeto = new InformacionDeEntrada( clave, objeto );
            this.informacionEntrada[ indice ].insertarAlFinal(nuevoObjeto);
            this.numeroDatos+=1;
            //para verificar que el factor de carga es tolerable <= 0.75
               this.rehash( );
            return nuevoObjeto.getObjeto();
        }
        else  // si la clave esta se encuentra en la tabla modificamos el objeto
             objetoAnterior.setObjeto( objeto);
    }
        return (T)objetoAnterior.getObjeto();
}
```

7.2. A continuación el Estudiante debe generar un Método en la clase **SimuladorTablaHash** del paquete **Mundo_TablaHash**, que realice el **llamado** al Método con el nuevo algoritmo creado en la Estructura de Datos **TablaHash**. Para el llamado debe utilizar el objeto creado en el Mundo **miTabla**. En este Método se deben insertar automáticamente los datos.

```
public void insertarDatos(){
    this.miTabla.insertar(204, "Uriel Garcia");
    this.miTabla.insertar(210, "Melany Rozo");
    this.miTabla.insertar(159, "Yulieth Pabon");
    this.miTabla.insertar(203, "Jorge Ochoa");
    this.miTabla.insertar(214, "Daniel Torres");
    this.miTabla.insertar(212, "Layne Granados");
    this.miTabla.insertar(250, "Cristian Osorio");
    this.miTabla.insertar(925, "Fabricio Mora");
    this.miTabla.insertar(268, "Jefferson Rangel");
    this.miTabla.insertar(725, "Absalon Vergara");
    this.miTabla.insertar(469, "Silvia Salazar");
    this.miTabla.insertar(147, "Wendy Hernandez");
}
```

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

7.3.1. Utilizando JavaFX SceneBuilder



7.3.2. Insertando directamente el elemento en Vista.fxml

```
<Button layoutX="212.0" layoutY="612.0" mnemonicParsing="false" onAction="#btCrear" prefWidth="98.0" text="Crear Tabla" />
```

Es importante resaltar, para ambos casos, que se debe asignar el evento **“OnAction”** del Button, para el ejemplo **“btCrearT”**, 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 **simtablahash** en la clase **Simulador** que permite realizar el llamado a la nueva funcionalidad.

```
@FXML
private void btCrearT(){
    this.simulador.insertarDatos();
    this.pintarTDA();
    this.impNota("Tabla creada automaticamente!", 0);
}
```

Además del llamado al Método creado en **SimuladorTablaHash**, el estudiante deberá invocar el método que le permita volver pintar la Tabla, 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:

Primero se crea la tabla con **11 Slots** para la inserció de datos:

Al ejecutar la nueva función el Simulador debe crear una Taba Hash con los datos insertados en esta automaticamente desde el codigo fuente:

"Se creó Tabla Hash con datos predeterminados".