MANUAL TECNICO

Presentado por
Grupo8

Marcos Arnoldo Itzep Ixmay 201907156
Josué Noel Mazariegos Gramajo 201900092
Byron Enrique Rumpich Sal 201907769
Carlos Alberto Rodas Galvez 201700420

Índice

GENERAL

En este manual se observan y se describen las funciones y procedimientos realizados para el funcionamiento correctos de este programa.

Clases del Paquete

- Main
- Configuracíon
- Main
- InicioUsuario
- Menú
- MenúUsu
- Usuario
- Files
- Serializacion

Configuracion

Variables

```
3       public class Configuracion {
4           public String name;
5           public String address;
6           public int phone;
7           public String load;
```

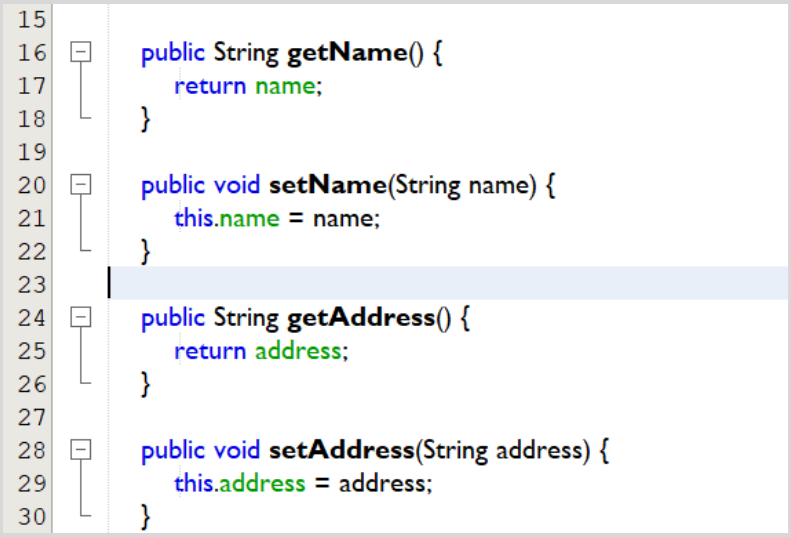
Estas variables son los atributos de la clase Configuración que corresponden a los datos del establecimiento.

Asignaciones

```
9       public Configuracion(String name, String address, int phone, String load) {
10          this.name = name;
11          this.address = address;
12          this.phone = phone;
13          this.load = load;
14      }
```

Métodos

Estos métodos se utilizan para mostrar o modificar los atributos.



```
15
16      public String getName() {
17          return name;
18      }
19
20      public void setName(String name) {
21          this.name = name;
22      }
23
24      public String getAddress() {
25          return address;
26      }
27
28      public void setAddress(String address) {
29          this.address = address;
30      }
```

toString

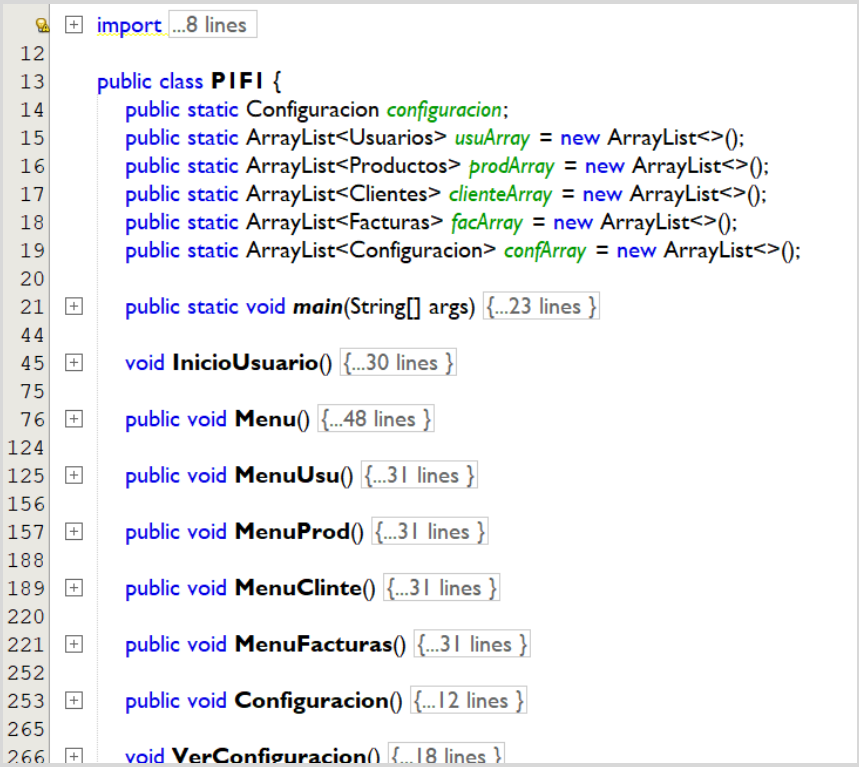
Este método se utiliza para convertir cualquier objeto en una cadena de texto

```
@Override
public String toString(){
    return "Name:"+name+"\tAddres:"+address+"\tPhone:"+phone+"\tLoad:"+load+ "\n";
}
```

****Estos bloques se replican en los demás objetos con sus respectivos atributos.**

Main

Dentro de esta clase están los métodos encargados del funcionamiento del programa.



```
import ...8 lines

public class PIFI {
    public static Configuracion configuracion;
    public static ArrayList<Usuarios> usuArray = new ArrayList<>();
    public static ArrayList<Productos> prodArray = new ArrayList<>();
    public static ArrayList<Clientes> clienteArray = new ArrayList<>();
    public static ArrayList<Facturas> facArray = new ArrayList<>();
    public static ArrayList<Configuracion> confArray = new ArrayList<>();

    public static void main(String[] args) {...23 lines }

    void InicioUsuario() {...30 lines }

    public void Menu() {...48 lines }

    public void MenuUsu() {...31 lines }

    public void MenuProd() {...31 lines }

    public void MenuClinte() {...31 lines }

    public void MenuFacturas() {...31 lines }

    public void Configuracion() {...12 lines }

    void VerConfiguracion() {...18 lines }
```

InicioUsuario

```
45  void InicioUsuario() {
46      System.out.println("===Inicio de Sesión===");
47      System.out.println("Introduzca su Usuario");
48      System.out.print(">>>");
49      Scanner lec = new Scanner(System.in);
50      String usuario = lec.nextLine();
51
52      System.out.println("Introduzca su Contraseña");
53      System.out.print(">>>");
54      Scanner lect = new Scanner(System.in);
55      String contraseña = lec.nextLine();
56
57      boolean entrar_menu = true;
58      if (entrar_menu == true) {
59          for (int i = 0; i < usuArray.size(); i++) {
60              if (usuArray.get(i).getUsername().equals(usuario) && usuArray.get(i).getPassword().equals(contraseña)) {
61                  System.out.println("Bienvenido al Menu");
62                  Errors.addToEndFile("errors.log","\t"+new Date()+"\tInicio de sesión exitoso"+"\n");
63                  Menu();
64                  entrar_menu = false;
65
66              }
67          }
68      }if (entrar_menu){
69          Errors.addToEndFile("errors.log","\t"+new Date()+"\tInicio de sesión Fallido"+"\n");
70          System.out.println("Usuario o contraseña incorrectos");
71          InicioUsuario();
72      }
73
74      }
```

El siguiente método permite a los usuarios ingresar con usuarios precargados por medio de una contraseña.

Menu

```
75
76      public void Menu() {
77          boolean entrar_menu = true;
78          while (entrar_menu == true) {
79              System.out.println("===Seleccione lo que desea cargar===");
80              System.out.println(" 1. Información del Restaurante");
81              System.out.println(" 2. Usuarios ");
82              System.out.println(" 3. Productos");
83              System.out.println(" 4. Clientes");
84              System.out.println(" 5. Facturas");
85              System.out.println(" 6. Guardar Cambios");
86              System.out.println(" 7. Salir");
87              System.out.println("====================================");
88              System.out.print(">>>Escribir el número de opción: >");
89
90              Scanner lec = new Scanner(System.in);
91              int opccion = lec.nextInt();
92              while (opccion < 1 || 7 < opccion) {
93                  System.out.println(">>>Error, Escribir una opción del 1 al 7>>>");
94                  opccion = lec.nextInt();
95              }
96              switch (opccion) {
97                  case 1:
98                      VerConfiguracion();
99                      break;
100                 case 2:
101                     MenuUsu();
102                     break;
103                 case 3:
104                     MenuProd();
105                     break;
```

Si el usuario ingresa como administrador este es el menú que imprimirá el método ***Menú***, seguido de un condicional *swich* encargado de operar el método seleccionado a través de cada uno de los case y la selección del administrador.

MenuUsu

```
125    public void MenuUsu() {
126        boolean entrar_menu = true;
127        while (entrar_menu == true) {
128        System.out.println("===Seleccione lo que desea hacer===");
129        System.out.println(" 1. Listar Usuarios");
130        System.out.println(" 2. Eliminar Usuarios ");
131        System.out.println(" 3. Ver Usuario");
132        System.out.println(" 4. Volver al Menu");
133        System.out.println("====================================");
134        System.out.print(">>>Escribir el número de opción: >");
135        Scanner lec = new Scanner(System.in);
136        int opccion = lec.nextInt();
137
138        switch (opccion) {
139
140            case 1:
141                VerUsuarios();
142                break;
143            case 2:
144                EliminarUsuario();
145                break;
146            case 3:
147                BuscarUsuario();
148                break;
149            case 4:
150                Menu();
151                break;
152        }
153        }
154
155    }
```

Si el usuario ingresa al menú de usuarios un condicional *swich* encargado de operar el método seleccionado a través de cada uno de los case y la selección del administrador.

****Estos bloques se replican en los demás objetos con sus respectivos atributos.**

Usuarios

```
284
285     void Usuarios() {
286
287         String contenidousu = Files.getContentOfFile("C:\\Users\\lorena\\Desktop\\\\Chalio\\P1F1\\jsons\\users.json");
288         Gson gson = new Gson();
289         Usuarios[] usuarios  = gson.fromJson(contenidousu, Usuarios[].class);
290
            for (Usuarios usuario : usuarios) {
292             usuArray.add(usuario);
293         }
294
295     }
```

VerUsuarios

```
296
297     void VerUsuarios() {
298
299         boolean confirmar = true;
300         if (confirmar==true) {
301         for (int i = 0; i < usuArray.size(); i++) {
302             if (usuArray.get(i).getUsername() != null) {
303                 System.out.println("");
304                 System.out.println("Nombre del usuario : " + usuArray.get(i).getUsername());
305                 System.out.println("");
306                 confirmar = false;
307             }
308         }
309
310         }if (confirmar){
311             System.out.println("No se encontró el usuario");
312             MenuUsu();
313         }
314     }
```

EliminarUsuario

```
316    public void EliminarUsuario() {
317
318        System.out.println("Introduzca el User del usuario que desea eliminar");
319        System.out.print(">>>");
320        Scanner lect = new Scanner(System.in);
321        String usuario = lect.nextLine();
322        boolean confirmar = true;
323        if (confirmar==true) {
324
325            for (int i = 0; i < usuArray.size(); i++) {
326                if (usuArray.get(i).getUsername().equals(usuario)) {
327                    usuArray.remove(i);
328                    System.out.println("Usuario eliminado");
329                    Errors.addToEndFile("errors.log","\t"+new Date()+"\tUSERS: Se eliminó el usuario: "+usuario+"\n");
330                    confirmar = false;
331                }
332            }
333
334        }if (confirmar){
335            System.out.println("No se encontró el usuario");
336            MenuUsu();
337        }
338            //while (confirmar == false) {
339              //  System.out.println("No se encontró el usuario");
340                //confirmar = true;
341            }
```

Files

Este método se utiliza para la lectura del documento

```
7   public class Files {
8       public static String getContentOfFile(String pathname) {
9           File archivo = null;
10          FileReader fr = null;
11          BufferedReader br = null;
12
13          try {
14              // Apertura del fichero y creacion de BufferedReader para poder
15              // hacer una lectura comoda (disponer del metodo readLine()).
16              archivo = new File(pathname);
17              fr = new FileReader(archivo);
18              br = new BufferedReader(fr);
19              // Lectura del fichero
20              String content = "";
21              String linea;
22              while ((linea = br.readLine()) != null) {
23                  content += linea + "\n";
24              }
25              return content;
26          } catch (Exception e) {
27              e.printStackTrace();
28          } finally {
29              // En el finally cerramos el fichero, para asegurarnos
30              // que se cierra tanto si todo va bien como si salta
31              // una excepcion.
32              try {
33                  if (null != fr) {
34                      fr.close();
35                  }
36              } catch (Exception e2) {
37                  e2.printStackTrace();
38              }
39          }
40          return "";
41      }
```

Serialización

```
7   public class Files {
8       public static String getContentOfFile(String pathname) {...34 lines }
42
43      public static void serialize(String pathname, Object object) {
44          // Serializar un objeto
45          try {
                ObjectOutputStream objectOutputStream = new ObjectOutputStream(new FileOutputStream(pathname));
47              objectOutputStream.writeObject(object);
48              objectOutputStream.close();
49          } catch (IOException e) {
                e.printStackTrace();
51          }
52      }
53      public static Object deserialize(String pathname) {
54          // Leer un objeto serializado
55          try {
                ObjectInputStream objectInputStream = new ObjectInputStream(new FileInputStream(pathname));
57              Object data = objectInputStream.readObject();
58              objectInputStream.close();
59              return data;
60          } catch (IOException | ClassNotFoundException e) {
                e.printStackTrace();
62          }
63          return null;
64      }
65
66  }
67
```