Universidad de Panamá
Facultad de Informática, Electrónica y Comunicación
Centro Regional Universitario de Veraguas

Laboratorio
CLASE SCANNER

Autor
Faustino Aguilar Quirós
2-732-727

Facilitador
Prof. Abundio Mendoza

II° Semestre de 2015

DESARROLLO DE LA ESTRATEGIA

En este proyecto se utilizó el lenguaje de programación Java para desarrollar un programa de computadora que simulara un banco.

En un banco hay cuentahabientes que realizan depósitos y retiros dependiendo de las condiciones de la cuenta.

Problema: usted debe crear una clase que denominaremos Banco cuyas propiedades son nombre de cliente , saldo inicial y final de la cuenta. El comportamiento de la clase serán las actividades a) depósito, b) retiro, c) verifica saldo, que servirá para saber si se puede o no hacer un retiro, d)despliega nombre del cliente y saldo.

Después de conocer la clase Scanner, usted puede elaborar una clase principal que crea objetos del Banco(clientes con nombre y saldo inicial) y que lea n transaciones de retiro y depósito del cliente (objeto de Banco) y despliegue el saldo inicial y el saldo final de cada cuentahabiente. Si se trata de un retiro se debe verificar si la transacción se puede realizar.

GLOSARIO

Abstracción: Razonamiento para crear las clases capaces de dar la solución al planteamiento.

Atributo: Datos y propiedades del objeto.

Clase: Propiedad y comportamiento de un tipo de objeto concreto.

Constructor: es el método privado que se ejecuta al realizar la instancia de una clase.

Evento: Acción que provoca una interacción del usuario con la máquina.

Herencia: Una clase puede heredar los campos y métodos de otra clase.

Instancia: Lectura de los datos y funciones de una clase y la creación de un obj a partir de éstos.

Mensaje: La comunicación dirigida a un objeto ordenándole que ejecute uno de sus métodos.

Método: Algoritmo asociado a un objeto. Define aquello que el objeto puede hacer

Objeto: Entidad provista de un conjunto de atributos (datos) y de comportamiento o funcionalidad (métodos). Es una instancia a una clase.

Polimorfismo: Varios objetos de la misma clase pueden actuar de diferente manera.

Código Fuente

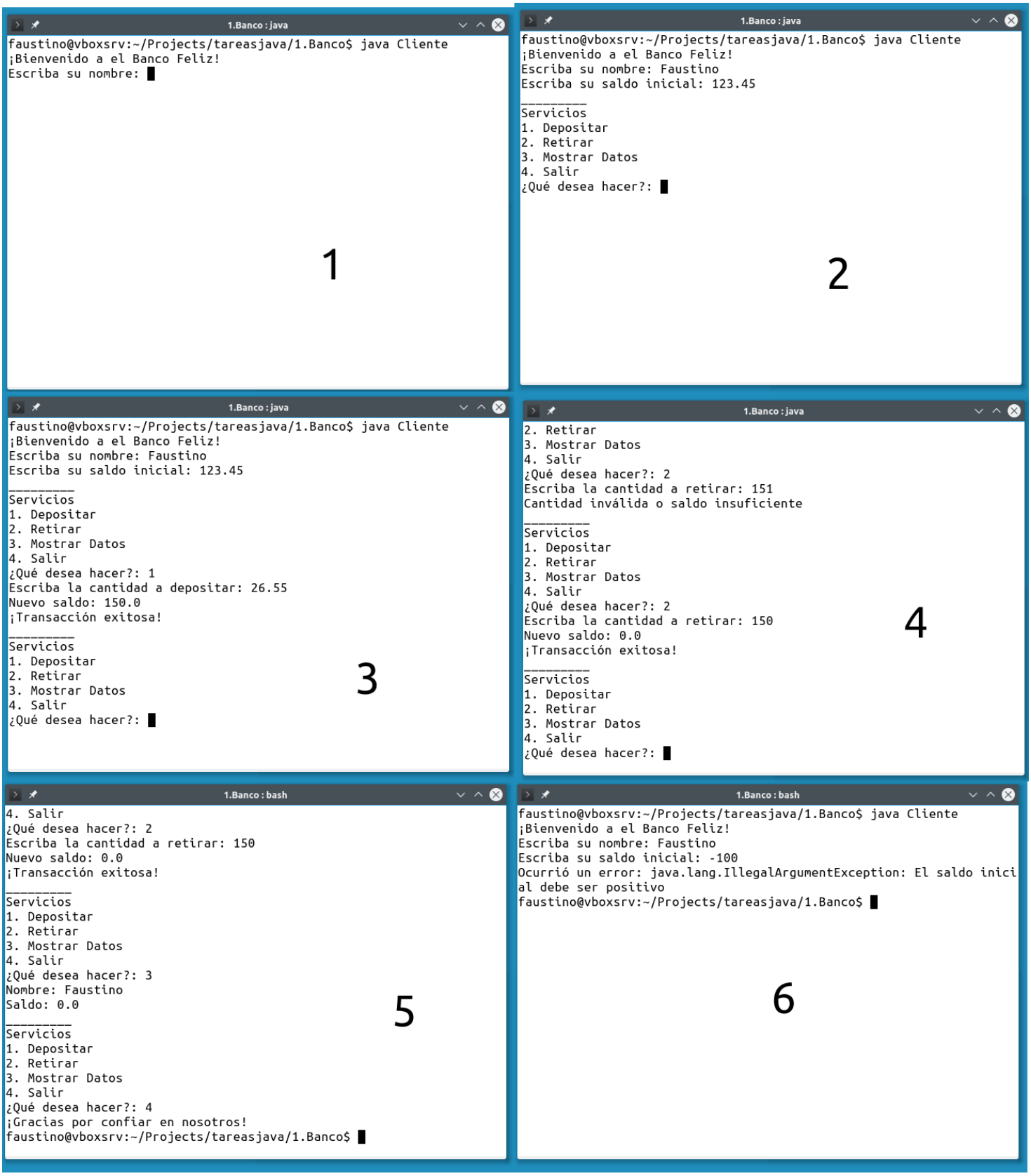
Banco.java

```
import java.io.*;

class Banco {
  /// Clase que contiene los métodos y atributos
  // del cuentahabiente.
  private String cliente;
  private float saldo;

  // Constructor de la Clase Banco
  Banco(String cliente, float saldo) {
    this.cliente = cliente;
    this.saldo = saldo;
    // Lanza una excepción cuando el saldo inicial es negativo
    if (!verificaSaldo(saldo)) {
      throw new IllegalArgumentException("El saldo inicial debe ser positivo");
    }
  }

  // deposito, retiro y verifica saldo retornan valores
  // de verdadero o falso
  public boolean deposito(float dinero) {
    // Solo deposita si la cantidad es válida
    if (dinero > 0) {
      this.saldo += dinero;
      System.out.println("Nuevo saldo: " + this.saldo);
      return true;
    } else {
      return false;
    }
  }
  public boolean retiro(float dinero){
    // Primero utiliza el método verificaSaldo
    if (this.verificaSaldo(dinero)) {
      this.saldo -= dinero;
      System.out.println("Nuevo saldo: " + this.saldo);
      return true;
    } else {
      return false;
    }
  }
  public boolean verificaSaldo(float dinero){
    // Verifica si el saldo es suficiente
    if (this.saldo >= dinero && dinero > 0) {
      return true;
    } else {
      return false;
    }
  }

  public void despliega(){
    // Muestra información sobre el cuentahabiente
    System.out.println("Nombre: " + this.cliente);
    System.out.println("Saldo: " + this.saldo);
  }
}
```

Cliente.java

```
import java.util.*;  // Contiene clase Scanner

class Cliente {
  // Clase principal que llama a la clase Banco
  public static void main(String[] args) {
    // Variables a utilizar
    int opcion = 4;
    String nombre;
    float saldo, dinero;
    Scanner scan = new Scanner(System.in);  // Maneja la entrada por teclado
    System.out.println("¡Bienvenido a el Banco Feliz!");
    // Utiliza try para manejar excepciones
    try {
      // Lee los datos del cuentahabiente
      System.out.print("Escriba su nombre: ");
      nombre = scan.nextLine();
      System.out.print("Escriba su saldo inicial: ");
      saldo = scan.nextFloat();
      // Instancia la clase Banco
      Banco cuenta = new Banco(nombre, saldo);
      do {
        // Menu con opciones que representan a
        // los métodos de la clase Banco
        System.out.println("_________\nServicios");
        System.out.println("1. Depositar");
        System.out.println("2. Retirar");
        System.out.println("3. Mostrar Datos");
        System.out.println("4. Salir");
        System.out.print("¿Qué desea hacer?: ");
        opcion = scan.nextInt();
        if (opcion == 1) {
          System.out.print("Escriba la cantidad a depositar: ");
          dinero = scan.nextFloat();
          if (cuenta.deposito(dinero)) {
            System.out.println("¡Transacción exitosa!");
          } else {
            System.out.println("Cantidad inválida");
          };
        } else if (opcion == 2) {
          System.out.print("Escriba la cantidad a retirar: ");
          dinero = scan.nextFloat();
          if (cuenta.retiro(dinero)) {
            System.out.println("¡Transacción exitosa!");
          } else {
            System.out.println("Cantidad inválida o saldo insuficiente");
          };
        } else if (opcion == 3) {
          cuenta.despliega();
        }
      } while (opcion != 4);
      System.out.println("¡Gracias por confiar en nosotros!");
    } catch (Exception e) {
      System.out.println("Ocurrió un error: " + e);
    }
  }
}
```

EVIDENCIA


1.Banco : java
faustino@vboxsrv:~/Projects/tareasjava/1.Banco$ java Cliente
¡Bienvenido a el Banco Feliz!
Escriba su nombre:
1
1.Banco : java
faustino@vboxsrv:~/Projects/tareasjava/1.Banco$ java Cliente
¡Bienvenido a el Banco Feliz!
Escriba su nombre: Faustino
Escriba su saldo inicial: 123.45
____
Servicios
1. Depositar
2. Retirar
3. Mostrar Datos
4. Salir
¿Qué desea hacer?:
2
1.Banco : java
faustino@vboxsrv:~/Projects/tareasjava/1.Banco$ java Cliente
¡Bienvenido a el Banco Feliz!
Escriba su nombre: Faustino
Escriba su saldo inicial: 123.45
____
Servicios
1. Depositar
2. Retirar
3. Mostrar Datos
4. Salir
¿Qué desea hacer?: 1
Escriba la cantidad a depositar: 26.55
Nuevo saldo: 150.0
¡Transacción exitosa!
____
Servicios
1. Depositar
2. Retirar
3. Mostrar Datos
4. Salir
¿Qué desea hacer?:
3
1.Banco : java
2. Retirar
3. Mostrar Datos
4. Salir
¿Qué desea hacer?: 2
Escriba la cantidad a retirar: 151
Cantidad inválida o saldo insuficiente
____
Servicios
1. Depositar
2. Retirar
3. Mostrar Datos
4. Salir
¿Qué desea hacer?: 2
Escriba la cantidad a retirar: 150
Nuevo saldo: 0.0
¡Transacción exitosa!
____
Servicios
1. Depositar
2. Retirar
3. Mostrar Datos
4. Salir
¿Qué desea hacer?:
4
1.Banco : bash
4. Salir
¿Qué desea hacer?: 2
Escriba la cantidad a retirar: 150
Nuevo saldo: 0.0
¡Transacción exitosa!
____
Servicios
1. Depositar
2. Retirar
3. Mostrar Datos
4. Salir
¿Qué desea hacer?: 3
Nombre: Faustino
Saldo: 0.0
____
Servicios
1. Depositar
2. Retirar
3. Mostrar Datos
4. Salir
¿Qué desea hacer?: 4
¡Gracias por confiar en nosotros!
faustino@vboxsrv:~/Projects/tareasjava/1.Banco$
5
1.Banco : bash
faustino@vboxsrv:~/Projects/tareasjava/1.Banco$ java Cliente
¡Bienvenido a el Banco Feliz!
Escriba su nombre: Faustino
Escriba su saldo inicial: -100
Ocurrió un error: java.lang.IllegalArgumentException: El saldo inici
al debe ser positivo
faustino@vboxsrv:~/Projects/tareasjava/1.Banco$
6

EVIDENCIA

En las capturas anteriores podemos ver las diferentes opciones que logra ejecutar nuetsro programa. A continuación la explicación de cada una:

1. Primera: presenta el banco llamado banco Feliz y luego pide el nombre del cliente.
2. Segunda: pide el saldo inicial del cuentahabiente y presenta un menú de opciones con las diferentes acciones que el cliente puede realizar.
3. Tercera: se realiza un depósito.
4. Cuarta: luego se trata de hacer un retiro sobre el saldo y el programa responde que la transacción es incorrecta ya que el saldo es insuficiente.
5. Quinta: se realiza un retiro dentro de lo permitido y se pide que muestre información del cliente.
6. Sexta: se muestra un error cuando el cliente introduce un saldo inicial inválido.

Cada vez que se realiza una transacción exitosa de depósito o retiro se muestra el nuevo saldo al cliente y se presenta nuevamente el menú de opciones hasta que se elija la opción 4 para salir, mostrando un texto de agradecimiento.

COMENTARIOS

Es interezante como podemos utilizar Java ara organizar nuestro proyecto de una manera diferente a la que conocíamos con lenguajes estructurados como C, el paradigma orientado a objetos utilizados por muchos lenguajes ha sido una buena herramienta para realizar este trabajo.

También cabe destacar que con este proyecto logramos entender muchos conceptos básicos de la POO y de Java como lo es la encapsulación, uso de construcciones y métodos.

REFERENCIAS

1. Jasa2S, www.java2s.com/Tutorials/Java/java.util/Scanner/Java_Scanner_nextFloat_.htm, se utilizó para ver ejemplos sobre el uso de la clase Scanner en Java.
2. M. Linares, 2011, http://stackoverflow.com/questions/6942624/how-to-throw-a-general-exception-in-java, Stackoverflow, se utilizó para ver detalles sobre el manejo de excepciones en Java.
3. Apuntes de Clase de Programacion Orientada a Objetos del Profesor Abundio Mendoza.