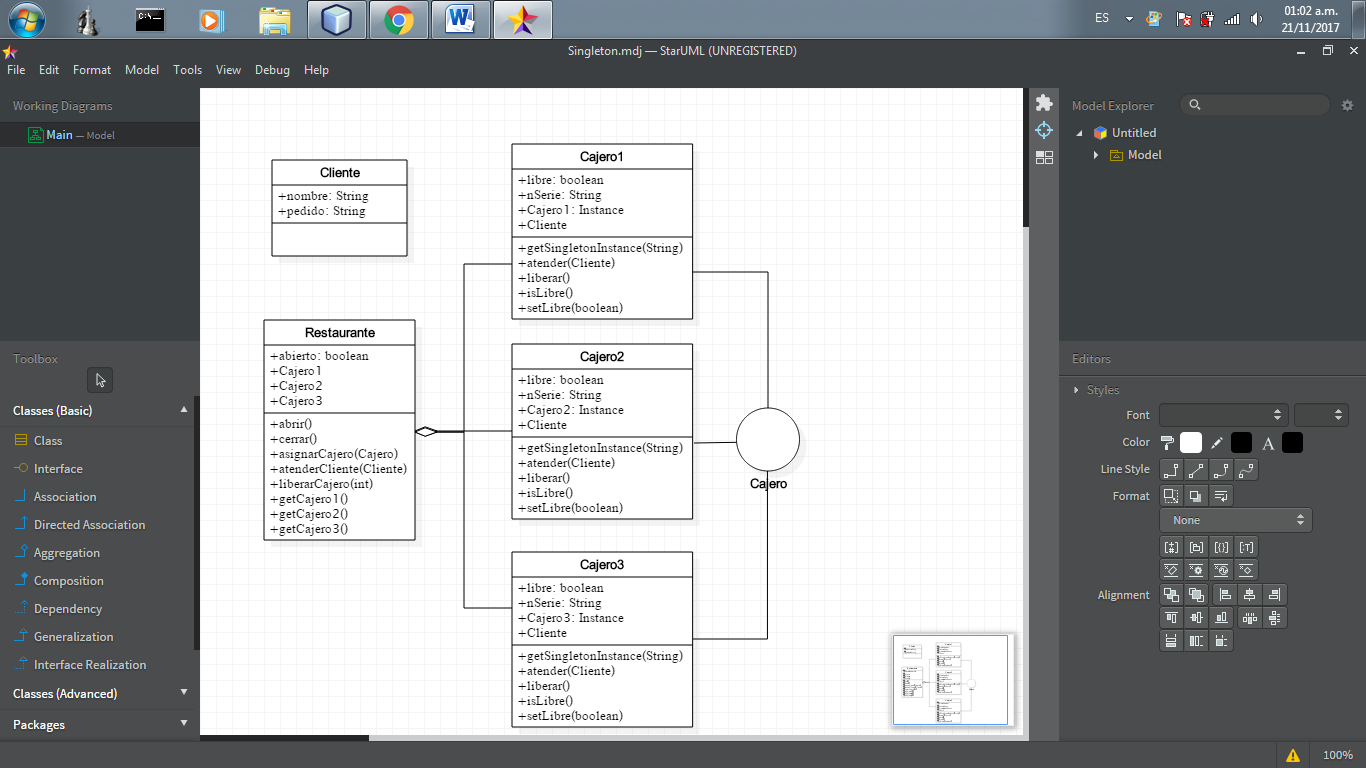
**ESCENARIO:**

Planteamos un escenario que sucedería dentro de un restaurante, la situación problema se presenta para el control del establecimiento de cajeros para atender en el restaurante, asegurando que exista una única instancia de cada cajero.



public class SingletonCajeros {

public static void main(String[] args) {

Restaurante restaurantePrueba = new Restaurante();

restaurantePrueba.abrir();

Cliente cliente;

String opcion;

int num;

while (restaurantePrueba.abierto == true) {

System.out.println("-------------------------");

System.out.println("BIENVENIDO AL RESTAURANTE");

System.out.println("\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_");

System.out.println("...CAJEROS DISPONIBLES...");

System.out.println("......................");

System.out.println("Cajero1: " + restaurantePrueba.getCajero1().isLibre());

System.out.println("Cajero2: " + restaurantePrueba.getCajero2().isLibre());

System.out.println("Cajero3: " + restaurantePrueba.getCajero3().isLibre());

System.out.println("......................");

opcion = JOptionPane.showInputDialog(null, "Digite Una Opcion:\n(1)NuevoCliente \n(2)LiberarCajero \n(3)Salir");

try {

switch (opcion) {

case "1":

String nombre = JOptionPane.showInputDialog(null, "Nombre Cliente: ");

cliente = new Cliente(nombre);

String orden = JOptionPane.showInputDialog(null, "Orden: ");

cliente.setPedido(orden);

System.out.println("Atendiendo...Cajero#"+restaurantePrueba.atenderCliente(cliente));

System.out.println("Cliente: "+nombre+"");

System.out.println("Orden: "+orden+"");

break;

case "2":

try {

num = Integer.parseInt(JOptionPane.showInputDialog(null, "Liberar Cajero #: "));

restaurantePrueba.liberarCajero(num);

System.out.println("Cajero #" + num + "Liberado...");

} catch (NumberFormatException ex) {

System.out.println("No Es Un Valor Valido");

}

break;

case "3":

restaurantePrueba.cerrar();

break;

default:

System.out.println("Opcion NO VALIDA");

break;

}

} catch (NumberFormatException ex) {

System.out.println("No Es Un Valor Valido");

}

}

}

}

public class Restaurante {

private Cajero1 cajero1;

private Cajero2 cajero2;

private Cajero3 cajero3;

public boolean abierto;

public Restaurante() {

this.cajero1 = Cajero1.getSingletonInstance("1");

this.cajero2 = Cajero2.getSingletonInstance("2");

this.cajero3 = Cajero3.getSingletonInstance("3");

}

public void abrir() {

abierto = true;

cajero1.liberar();

cajero2.liberar();

cajero3.liberar();

}

public void cerrar() {

abierto = false;

}

public Cajero asignarCajero(Cajero cajero) {

return cajero;

}

public int atenderCliente(Cliente cliente) {

int cajero=0;

if (cajero1.getSingletonInstance("1").isLibre() == true) {

cajero1.getSingletonInstance("1").atender(cliente);

cajero1.getSingletonInstance("1").setLibre(false);

cajero = 1;

} else if (cajero2.getSingletonInstance("2").isLibre() == true) {

cajero2.getSingletonInstance("2").atender(cliente);

cajero2.getSingletonInstance("2").setLibre(false);

cajero = 2;

} else if (cajero3.getSingletonInstance("3").isLibre() == true) {

cajero3.getSingletonInstance("3").atender(cliente);

cajero3.getSingletonInstance("3").setLibre(false);

cajero = 3;

} else if (abierto = true) {

System.out.println("Cajeros Ocupados INTENTE LIBERAR ALGUNO...");

}

System.out.println(".........................");

return cajero;

}

public void liberarCajero(int num) {

if (num == 1) {

cajero1.liberar();

} else if (num == 2) {

cajero2.liberar();

} else if (num == 3) {

cajero3.liberar();

} else {

System.out.println("Cajero No Existe");

}

}

public Cajero1 getCajero1() {

return cajero1;

}

public Cajero2 getCajero2() {

return cajero2;

}

public Cajero3 getCajero3() {

return cajero3;

}

}

public interface Cajero {

public void atender(Cliente cliente);

public void liberar();

public boolean isLibre();

public void setLibre(boolean ocupado);

}

public class Cajero1 implements Cajero{

public boolean libre;

public String nSerie;

public static Cajero1 instance;

public Cliente cliente;

private Cajero1(String nSerie) {

this.nSerie = nSerie;

}

public static Cajero1 getSingletonInstance(String nSerie) {

if (instance == null) {

instance = new Cajero1(nSerie);

}

return instance;

}

@Override

public void atender(Cliente cliente) {

if(libre==true){

this.cliente=cliente;

}

}

@Override

public void liberar() {

setLibre(true);

}

@Override

public boolean isLibre() {

return libre;

}

@Override

public void setLibre(boolean libre) {

this.libre = libre;

}

}

public class Cajero2 implements Cajero{

public boolean libre;

public String nSerie;

public static Cajero2 instance;

public Cliente cliente;

private Cajero2(String nSerie) {

this.nSerie = nSerie;

}

public static Cajero2 getSingletonInstance(String nSerie) {

if (instance == null) {

instance = new Cajero2(nSerie);

}

return instance;

}

@Override

public void atender(Cliente cliente) {

if(libre==true){

this.cliente=cliente;

}

}

@Override

public void liberar() {

setLibre(true);

}

@Override

public boolean isLibre() {

return libre;

}

@Override

public void setLibre(boolean ocupado) {

this.libre = ocupado;

}

}

public class Cajero3 implements Cajero {

public boolean libre;

public String nSerie;

public static Cajero3 instance;

public Cliente cliente;

private Cajero3(String nSerie) {

this.nSerie = nSerie;

}

public static Cajero3 getSingletonInstance(String nSerie) {

if (instance == null) {

instance = new Cajero3(nSerie);

}

return instance;

}

@Override

public void atender(Cliente cliente) {

if (libre == true) {

this.cliente = cliente;

}

}

@Override

public void liberar() {

setLibre(true);

}

@Override

public boolean isLibre() {

return libre;

}

@Override

public void setLibre(boolean ocupado) {

this.libre = ocupado;

}

}

public class Cliente {

private final String nombre;

private String pedido;

public Cliente(String nombre) {

this.nombre = nombre;

}

public String getPedido() {

return pedido;

}

public void setPedido(String pedido) {

this.pedido = pedido;

}

}

RESULTADOS:

