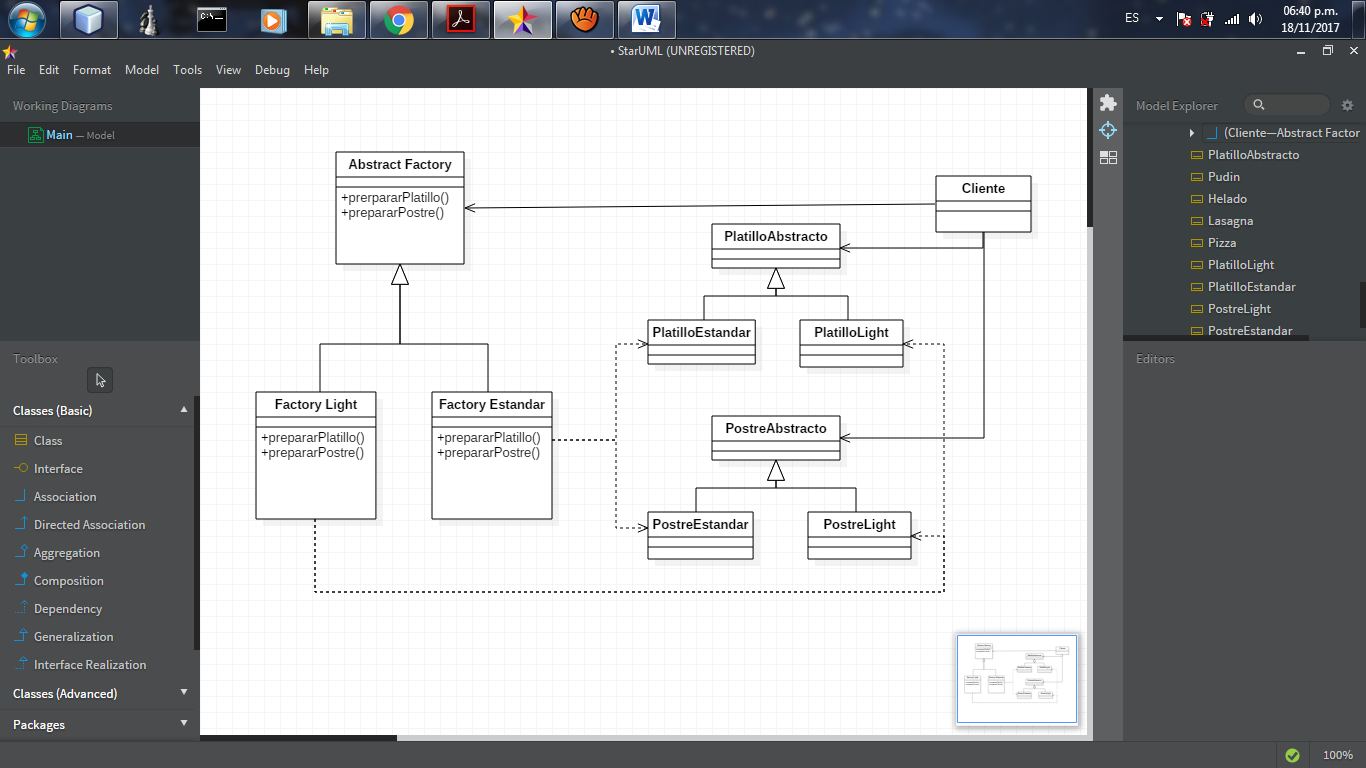
**ESCENARIO:**

Planteamos un escenario que sucedería dentro de un restaurante, la situación problema se presenta para la creación de distintos tipos de platos (normal o light) sin que se especifique en detalle la forma de su creación.



public class Cliente {

public static void main(String[] args) {

FabricaAbstracta fabrica;

boolean cerrar = false;

Vector<PlatilloAbstracto> platillosListos = new Vector<PlatilloAbstracto>();

Vector<PostreAbstracto> postresListos = new Vector<PostreAbstracto>();

while (cerrar == false) {

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

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

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

String orden = JOptionPane.showInputDialog(null, "Orden: (1)Platillo (2)Postre: ");

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

String tipo = JOptionPane.showInputDialog(null, "Tipo De Preparacion: (1)Normal (2)Light :");

int cantidad = Integer.parseInt(JOptionPane.showInputDialog(null, "Cantidad: # "));

switch (tipo) {

case "1":

fabrica = new FabricaEstandar();

if ("1".equals(orden)) {

while (cantidad > 0) {

platillosListos.addElement(fabrica.crearPlatillo(nombre));

System.out.println("Platillo Listo: " + nombre);

cantidad--;

}

}

if ("2".equals(orden)) {

while (cantidad > 0) {

postresListos.addElement(fabrica.crearPostre(nombre));

System.out.println("Postre Listo: " + nombre);

cantidad--;

}

}

break;

case "2":

fabrica = new FabricaLight();

if ("1".equals(orden)) {

while (cantidad > 0) {

platillosListos.addElement(fabrica.crearPlatillo(nombre));

System.out.println("Platillo Listo: " + nombre+"-Light");

cantidad--;

}

}

if ("2".equals(orden)) {

while (cantidad > 0) {

postresListos.addElement(fabrica.crearPostre(nombre));

System.out.println("Postre Listo: " + nombre+"-Light");

cantidad--;

}

}

break;

default:

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

break;

}

String exit = JOptionPane.showInputDialog("Nueva Orden: (1)SI (2)NO");

cerrar = !"1".equals(exit);

}

}

}

public abstract class FabricaAbstracta {

public abstract PlatilloAbstracto crearPlatillo(String nombre);

public abstract PostreAbstracto crearPostre(String nombre);

}

public abstract class PlatilloAbstracto {

protected String nombre;

public abstract String getNombre();

}

public abstract class PostreAbstracto {

protected String nombre;

public abstract String getNombre();

}

public class FabricaEstandar extends FabricaAbstracta {

@Override

public PlatilloAbstracto crearPlatillo(String nombre) {

return new PlatilloEstandar(nombre);

}

@Override

public PostreAbstracto crearPostre(String nombre) {

return new PostreEstandar(nombre);

}

}

public class FabricaLight extends FabricaAbstracta{

@Override

public PlatilloAbstracto crearPlatillo(String nombre) {

return new PlatilloLight(nombre);

}

@Override

public PostreAbstracto crearPostre(String nombre) {

return new PostreLight(nombre);

}

}

public class PlatilloEstandar extends PlatilloAbstracto {

public PlatilloEstandar(String nombre) {

this.nombre = nombre;

}

@Override

public String getNombre() {

return nombre;

}

}

public class PlatilloLight extends PlatilloAbstracto {

public PlatilloLight(String nombre) {

this.nombre = nombre;

}

@Override

public String getNombre() {

return nombre;

}

}

public class PostreEstandar extends PostreAbstracto {

public PostreEstandar(String nombre) {

this.nombre = nombre;

}

@Override

public String getNombre() {

return nombre;

}

}

public class PostreLight extends PostreAbstracto {

public PostreLight(String nombre) {

this.nombre = nombre;

}

@Override

public String getNombre() {

return nombre;

}

}

RESULTADOS:

