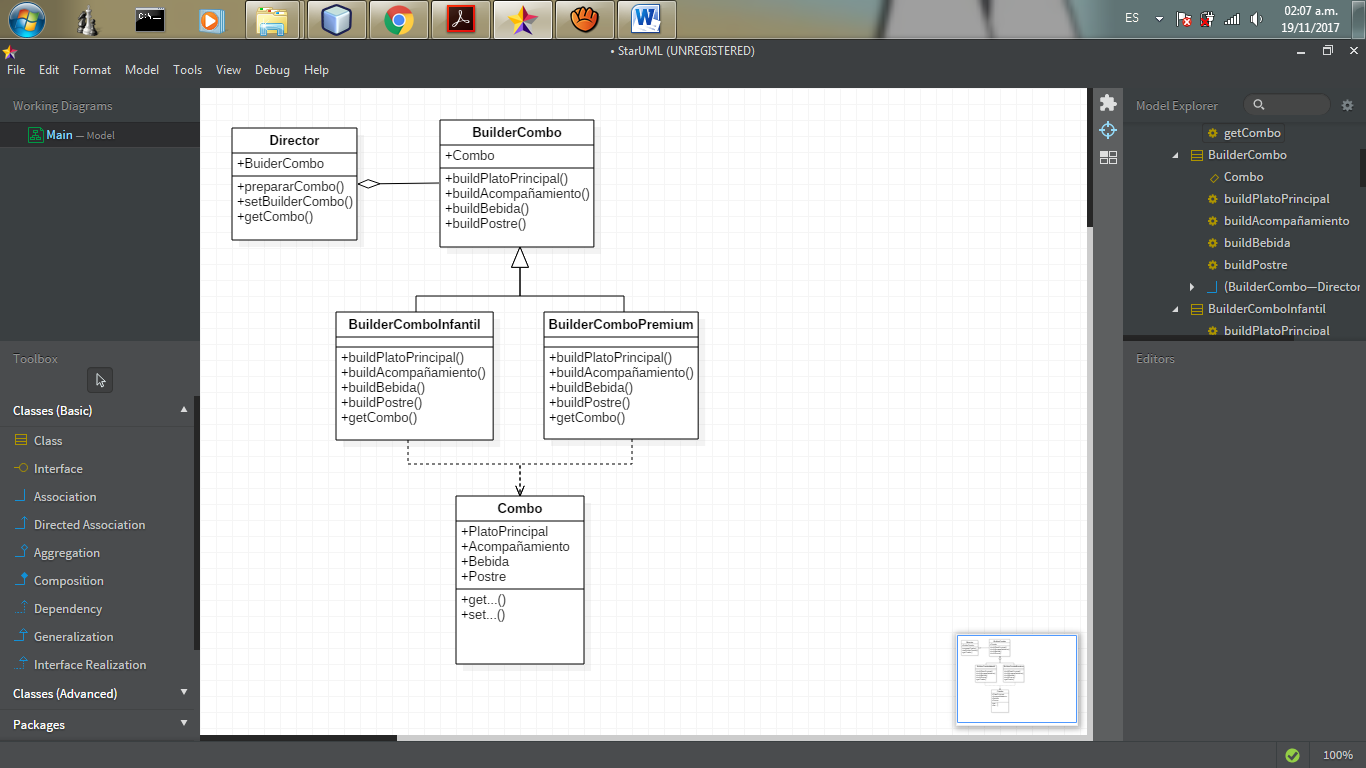
**ESCENARIO:**

Planteamos un escenario que sucedería dentro de un restaurante, la situación problema se presenta para la creación de distintos tipos de combos (Premium, Infantil) a su vez dichos combos están conformados por objetos y solo el director es quien sabe cómo preparar el combo.



public class Cliente {

public static void main(String[] args) {

boolean cerrar = false;

String tipoCombo;

DirectorCocina director = new DirectorCocina();

Combo combo;

while (cerrar == false) {

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

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

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

tipoCombo = JOptionPane.showInputDialog(null, "Tipo De Combo: (1)Premium (2)Infantil: ");

switch (tipoCombo) {

case "1":

director.setBuilderCombo(new BuilderComboPremium());

director.prepararCombo();

combo = director.getCombo();

System.out.println("PREPARADO COMBO PREMIUM");

System.out.println("PLATO PRINCIPAL: " + (combo.getPlatoPrincipal().getNombre())

+ "|| Tamaño: " + combo.getPlatoPrincipal().getTamaño());

System.out.println("ACOMPAÑAMIENTO: " + (combo.getAcompañamiento().getNombre())

+ "|| Tamaño: " + combo.getAcompañamiento().getTamaño());

System.out.println("BEBIDA: " + (combo.getBebida().getNombre())

+ "|| Tamaño: " + combo.getBebida().getTamaño());

System.out.println("POSTRE: " + (combo.getPostre().getNombre())

+ "|| Tamaño: " + combo.getPostre().getTamaño());

break;

case "2":

director.setBuilderCombo(new BuilderComboInfantil());

director.prepararCombo();

combo = director.getCombo();

System.out.println("PREPARADO COMBO INFANTIL");

System.out.println("PLATO PRINCIPAL: " + (combo.getPlatoPrincipal().getNombre())

+ "|| Tamaño: " + combo.getPlatoPrincipal().getTamaño());

System.out.println("ACOMPAÑAMIENTO: " + (combo.getAcompañamiento().getNombre())

+ "|| Tamaño: " + combo.getAcompañamiento().getTamaño());

System.out.println("BEBIDA: " + (combo.getBebida().getNombre())

+ "|| Tamaño: " + combo.getBebida().getTamaño());

System.out.println("POSTRE: " + (combo.getPostre().getNombre())

+ "|| Tamaño: " + combo.getPostre().getTamaño());

break;

default:

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

break;

}

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

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

}

}

}

public class DirectorCocina {

private BuilderCombo builderCombo;

public void prepararCombo() {

builderCombo.generarCombo();

builderCombo.buildPlatoPrincipal();

builderCombo.buildAcompañamiento();

builderCombo.buildBebida();

builderCombo.buildPostre();

}

public void setBuilderCombo(BuilderCombo builderCombo) {

this.builderCombo = builderCombo;

}

public Combo getCombo() {

return builderCombo.getCombo();

}

}

public abstract class BuilderCombo {

public Combo combo;

public Combo getCombo() {

return combo;

}

public void generarCombo(){

combo = new Combo();

}

public abstract void buildPlatoPrincipal();

public abstract void buildAcompañamiento();

public abstract void buildBebida();

public abstract void buildPostre();

}

public class BuilderComboInfantil extends BuilderCombo {

@Override

public void buildPlatoPrincipal() {

PlatoPrincipal platoP = new PlatoPrincipal();

platoP.setNombre("Hamburguesa Al Gusto");

platoP.setTamaño("mediano");

combo.setPlatoPrincipal(platoP);

}

@Override

public void buildAcompañamiento() {

Acompañamiento acompañamiento = new Acompañamiento();

acompañamiento.setNombre("Nuggets De Pollo");

acompañamiento.setTamaño("Medio");

combo.setAcompañamiento(acompañamiento);

}

@Override

public void buildBebida() {

Bebida bebida = new Bebida();

bebida.setNombre("Coca-Cola");

bebida.setTamaño("Pequeño");

combo.setBebida(bebida);

}

@Override

public void buildPostre() {

Postre postre = new Postre();

postre.setNombre("Helado De La Casa");

postre.setTamaño("Grande");

combo.setPostre(postre);

}

}

public class BuilderComboPremium extends BuilderCombo{

@Override

public void buildPlatoPrincipal() {

PlatoPrincipal platoPrincipal = new PlatoPrincipal();

platoPrincipal.setNombre("Bandeja 3 Carnes");

platoPrincipal.setTamaño("Medio");

combo.setPlatoPrincipal(platoPrincipal);

}

@Override

public void buildAcompañamiento() {

Acompañamiento acompañamiento = new Acompañamiento();

acompañamiento.setNombre("Ensalada Al Gusto");

acompañamiento.setTamaño("Medio");

combo.setAcompañamiento(acompañamiento);

}

@Override

public void buildBebida() {

Bebida bebida = new Bebida();

bebida.setTamaño("Grande");

bebida.setNombre("Jugo Al Gusto");

combo.setBebida(bebida);

}

@Override

public void buildPostre() {

Postre postre = new Postre();

postre.setNombre("Torta Del Dia");

postre.setTamaño("Medio");

combo.setPostre(postre);

}

}

public class Combo {

private PlatoPrincipal platoPrincipal;

private Acompañamiento acompañamiento;

private Bebida bebida;

private Postre postre;

public PlatoPrincipal getPlatoPrincipal() {

return platoPrincipal;

}

public void setPlatoPrincipal(PlatoPrincipal platoPrincipal) {

this.platoPrincipal = platoPrincipal;

}

public Acompañamiento getAcompañamiento() {

return acompañamiento;

}

public void setAcompañamiento(Acompañamiento acompañamiento) {

this.acompañamiento = acompañamiento;

}

public Bebida getBebida() {

return bebida;

}

public void setBebida(Bebida bebida) {

this.bebida = bebida;

}

public Postre getPostre() {

return postre;

}

public void setPostre(Postre postre) {

this.postre = postre;

}

}

RESULTADO:

El cliente elige el tipo de combo, y si desea ordenar mas…

