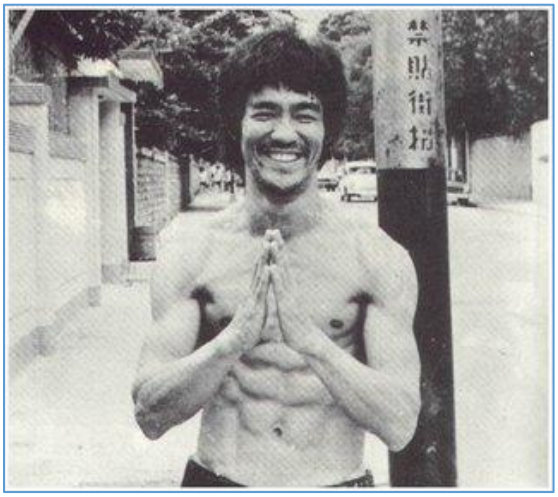
GUÍA DE LABORATORIO 03

ANTES DE INICIAR…

Como diría el legendario *Bruce Lee*…



Si tienes ***un vaso medio lleno*** no vas a poder recibir por completo toda la enseñanza que se quiere transmitir, por consiguiente, **"vaciarse" de todo lo que crees que sabes es lo más importante**, más que acumular conocimientos y luego simplemente rechazar lo nuevo creyendo que eso *"ya lo sabes"* …

Así que, de aquí en adelante, olvida todo lo que crees que sabes sobre objetos y empecemos de cero.

Un niño, cuando empieza a hablar, nos demuestra que ya entiende el concepto de "objetos", empieza a nombrar esas "cosas":

- "vaso",
- "agua",
- "papá",
- "mamá",
- "casa",
- "guau guau" ("perro"),
- etc.

Empecemos por un diálogo pequeño:

Micaela, de 5 años, dice: "mira el perro negro y blanco, se llama Tito, le toco la cabeza y mueve la cola, y si le doy de comer, al rato, hace popó".

¿OBJETOS?

¿ATRIBUTOS?

¿ACCIONES?

"Más importante que codificar es detectar los objetos sin preocuparnos -aún- del código que los representará" – EP

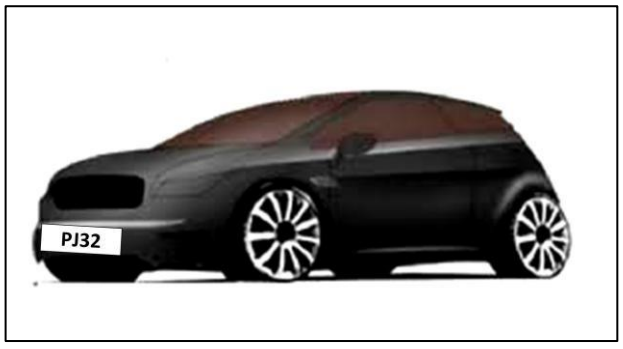
PROGRAMACIÓN ORIENTADA A OBJETOS

INTRODUCCION

La programación en lenguajes procedimentales –tales como BASIC, C, Pascal, Ada y COBOL- implica estructura de datos, diseño de algoritmos y traducción de algoritmos en código. La programación orientada a objetos es un tipo de programación que introduce construcciones específicas llamadas objetos que, a su vez, contienen datos y procedimientos. Los objetos hacen la programación más fácil y menos propensa a errores. C# es un lenguaje orientado a objetos donde los conjuntos o colecciones de objetos que cooperan entre si juegan un papel fundamental.

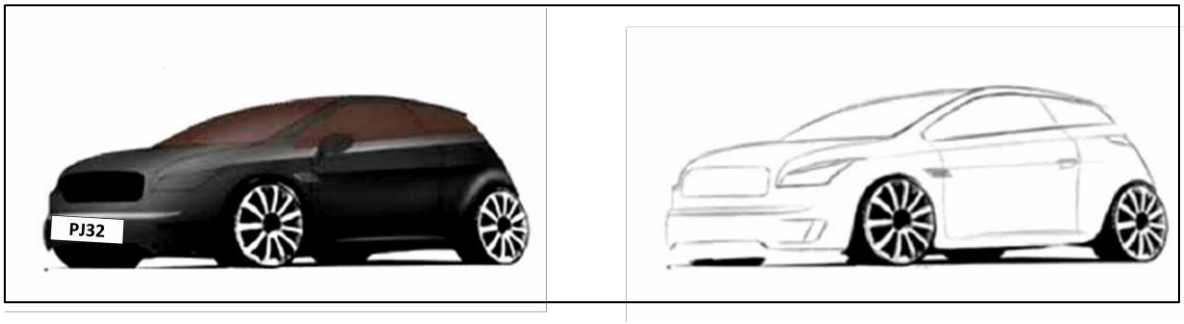
¿OBJETOS?

Es la representación concreta y detallada de algo en particular, tal representación determina su identidad (nombre único para distinguir un objeto de otro) su estado (conjunto de valores que caracterizan al objeto en un momento dado) y su comportamiento (conjunto de funciones que el objeto puede llevar a cabo).



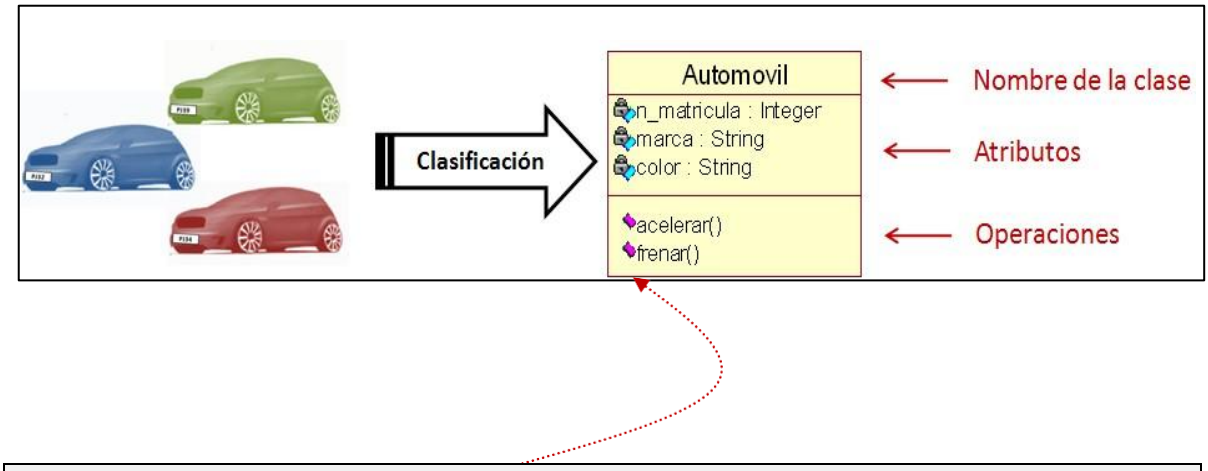
¿ABSTRACCION?

Consiste en capturar percibir y clasificar las características (datos distribuidos) y comportamientos (operaciones) necesarias (relevantes) del mundo real (proceso a sistematizar) para dar solución al problema.



¿CLASE?

Es la clasificación de las características y comportamientos comunes de objetos del mismo tipo. En la POO se dice que es una plantilla genérica para un conjunto de datos con las mismas características.



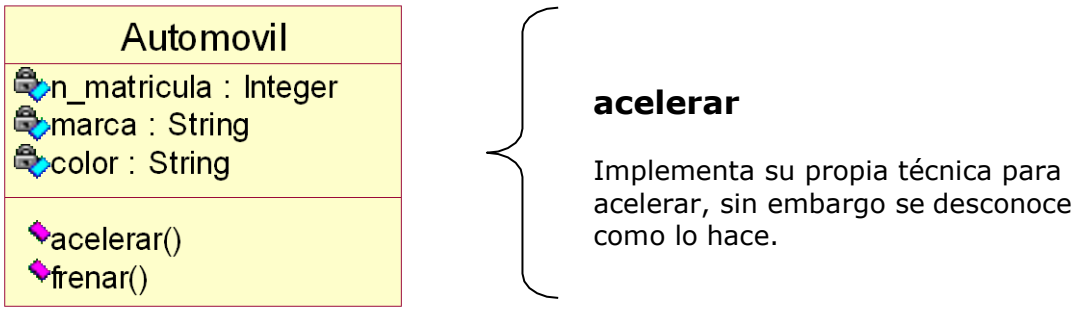
VISIBILIDAD
+Publico: Acceso interior y exterior, instancia (objeto) y herencia.
-Privado: Acceso interior, solo dentro de la clase.
#Protegido: Acceso interior y exterior solo por herencia.

Nota: Cuando un miembro (atributo-método), no tiene visibilidad entonces en Java es considerado como visibilidad del paquete.

CARACTERISTICAS DE LA POO

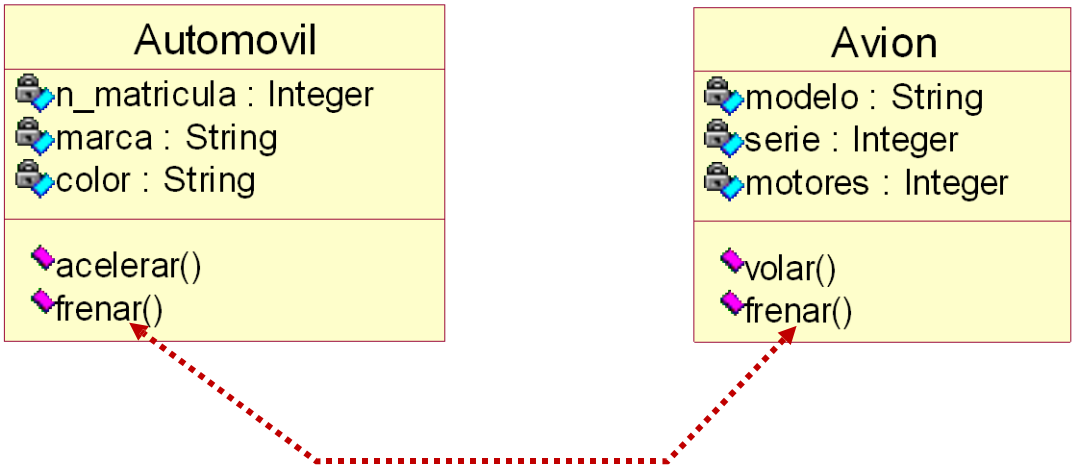
¿ENCAPSULAMIENTO?

Oculta la complejidad, es considerada como la caja negra solo se conoce su comportamiento pero no su detalle interno.



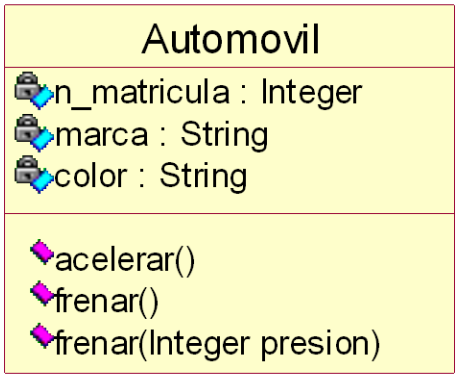
¿POLIMORFISMO?

Es la capacidad que tienen lo objetos de responder el mismo mensaje de diferentes formas, a través de un **método** (operación) con el mismo nombre pero con implementación diferente.



¿SOBRECARGA?

La sobrecarga en C# se puede definir varios métodos en un tipo con el mismo nombre pero con parámetros diferentes y el sistema selecciona el método que desea llamar examinando los parámetros.



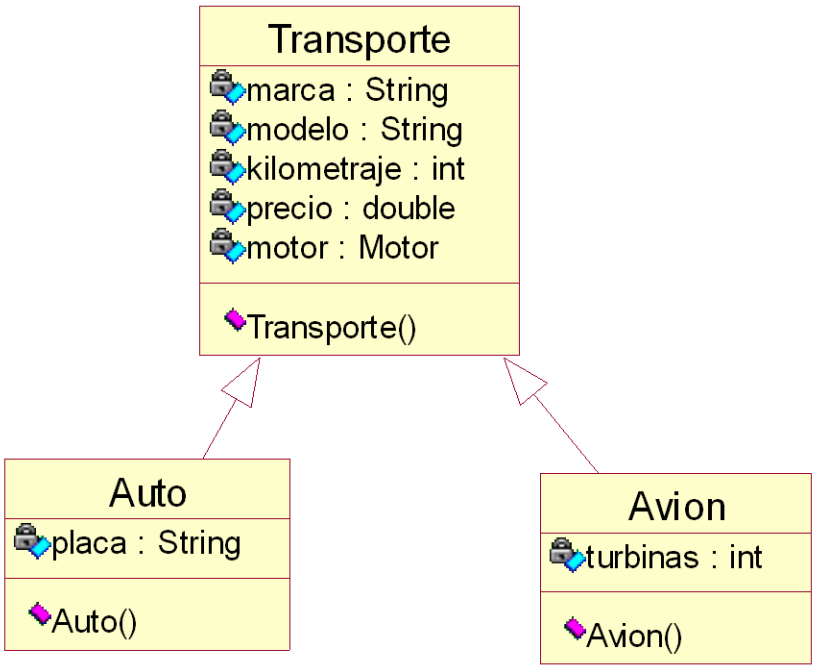
¿ASOCIACION?

Es una relación entre instancias de dos clases independientes entre ellas, además podemos decir que existe una asociación entre dos clases si una instancia de una clase debe conocer de la otra para poder ejecutar su trabajo.

¿HERENCIA?

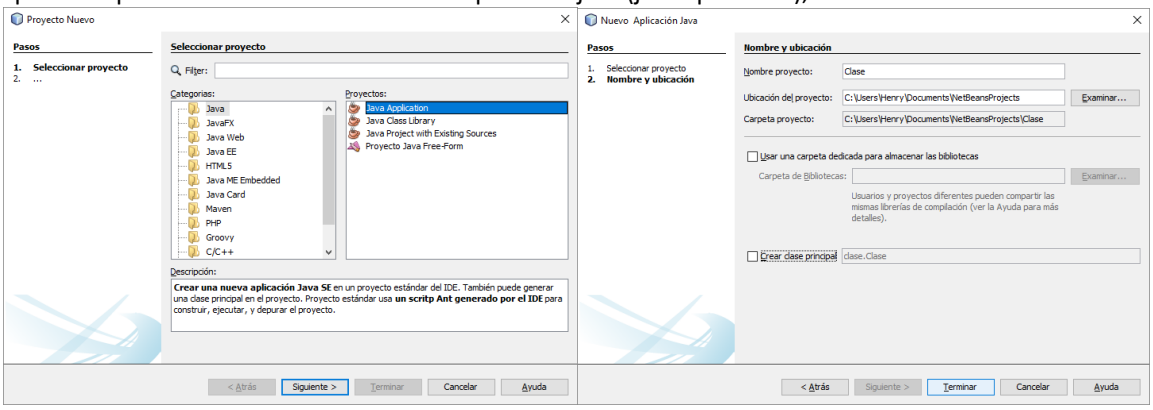
Es la característica más importante de la POO y permite utilizar objetos para construir nuevos objetos.

En la herencia se observa que existen clases genéricas (padre/superclase) que agrupan características y comportamientos similares para un conjunto de objetos y clases derivadas (hija/subclase) que extienden o redefinen la clase genérica.

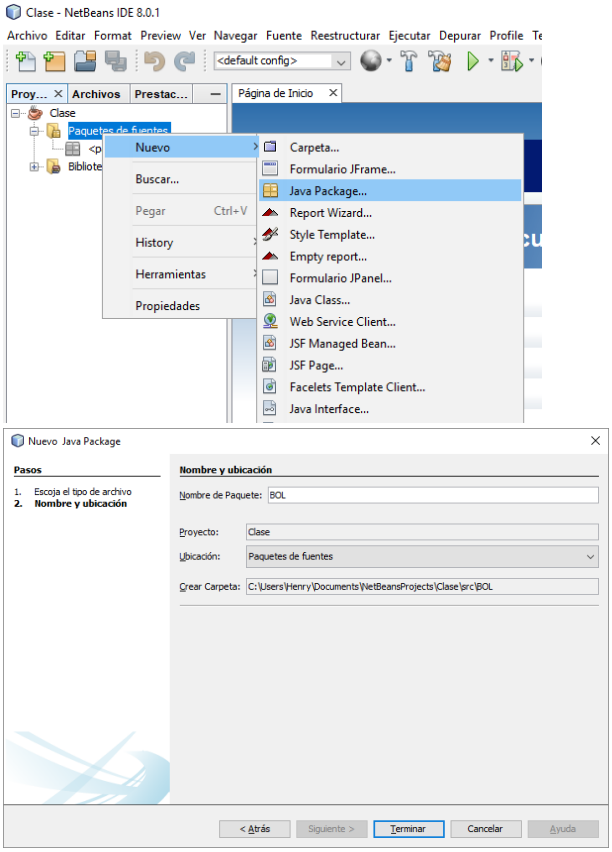


A continuación, trabajaremos con clases y objetos.

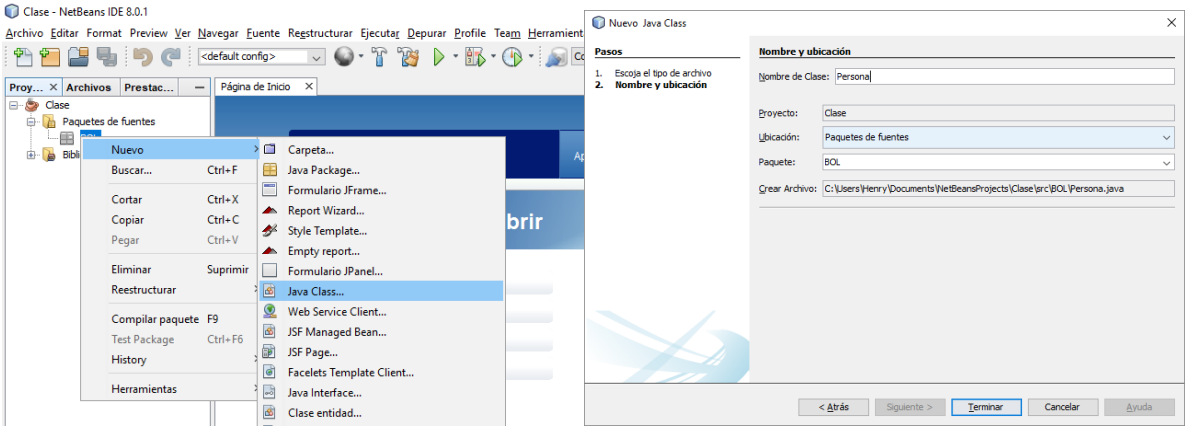
1. Lo primero qué debemos hacer es crear una aplicación java (java aplication), en el IDE NetBeans.



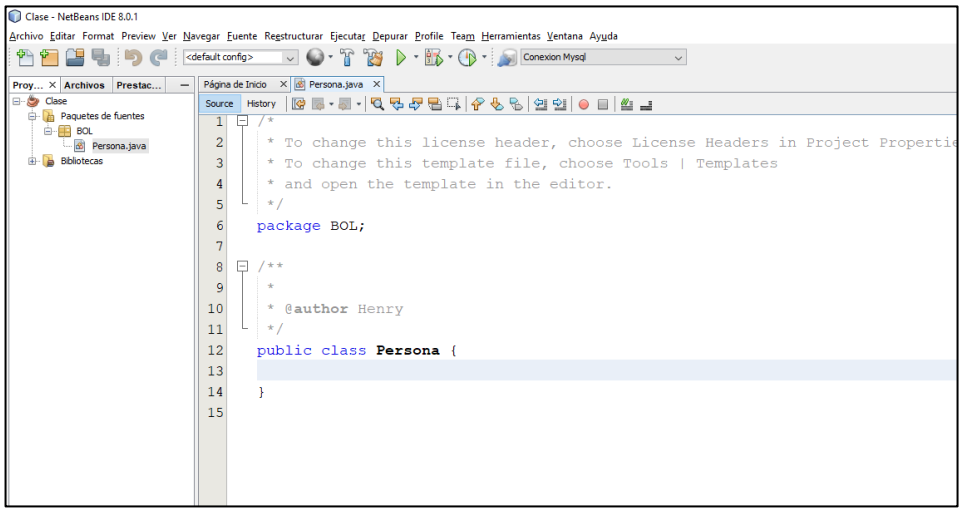
2. Creamos un paquete BOL (capa lógica de negocio), para alojar nuestras clases



3. Ahora creamos la clase Persona dentro del paquete BOL.



4. Dentro de la clase persona, entiende que es publica y además que el constructor esta creado implícitamente.



5. Creamos un campo de tipo cadena y con visibilidad privada.

```
private String nombres;
```

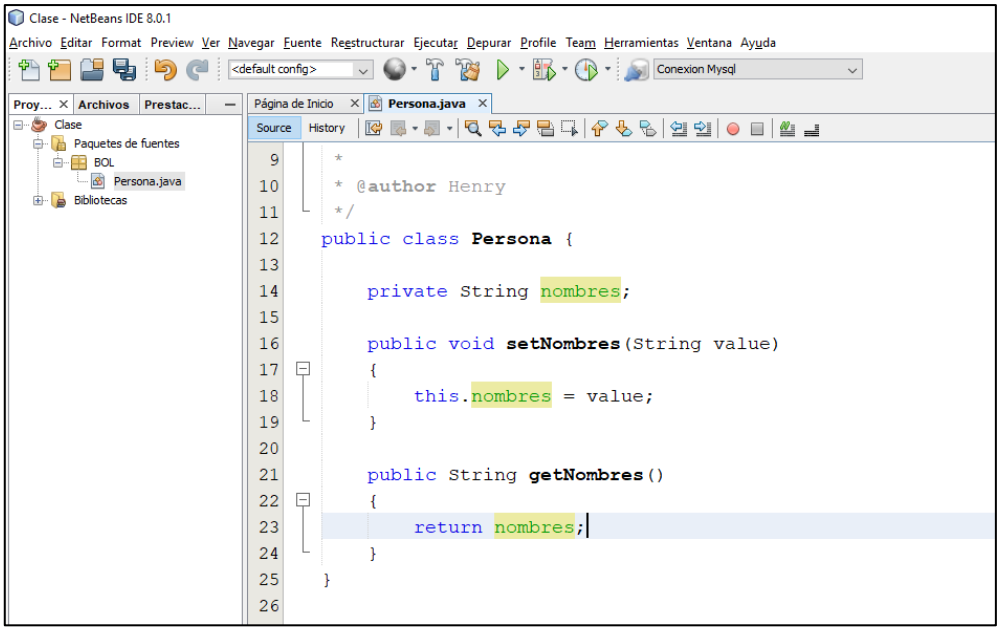
6. Para establecer valores a nuestro campo privado necesitamos de un método público como lo es nuestro SET, por tanto creamos "setNombres()".

```
public void setNombres(String value)
{
    this.nombres = value;
}
```

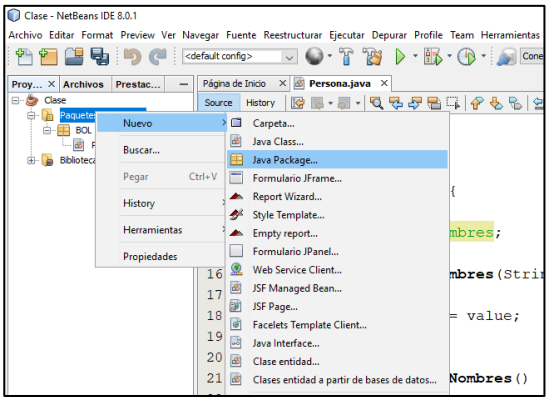
7. Para poder Obtener el valor que fue establecido al campo, necesitamos de los GET, en este caso crearemos el "getNombres()".

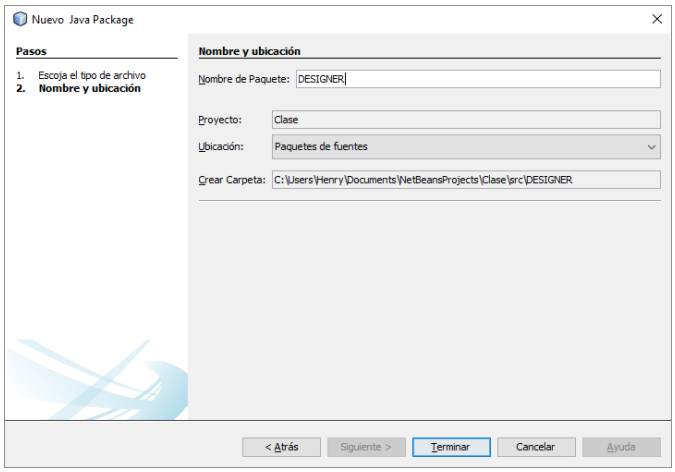
```
public String getNombres()
{
    return nombres;
}
```

8. La clase hasta el momento quedaría de la siguiente manera:

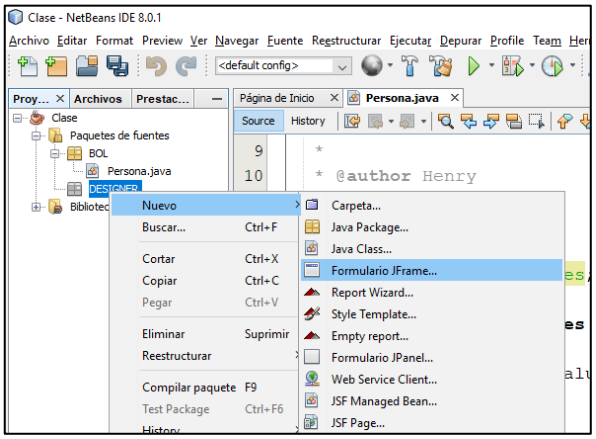


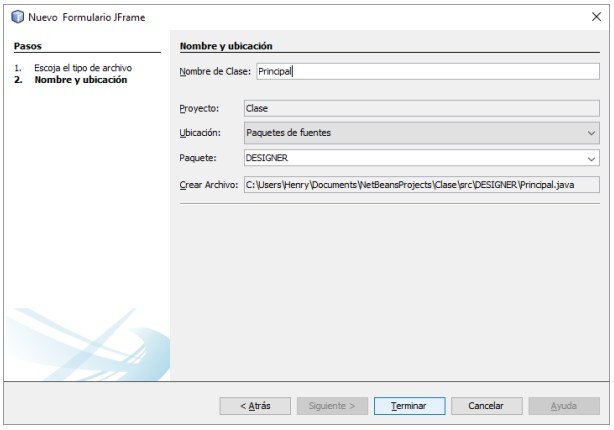
9. Ahora crearemos una capa de diseño que se llamara "DESIGNER"



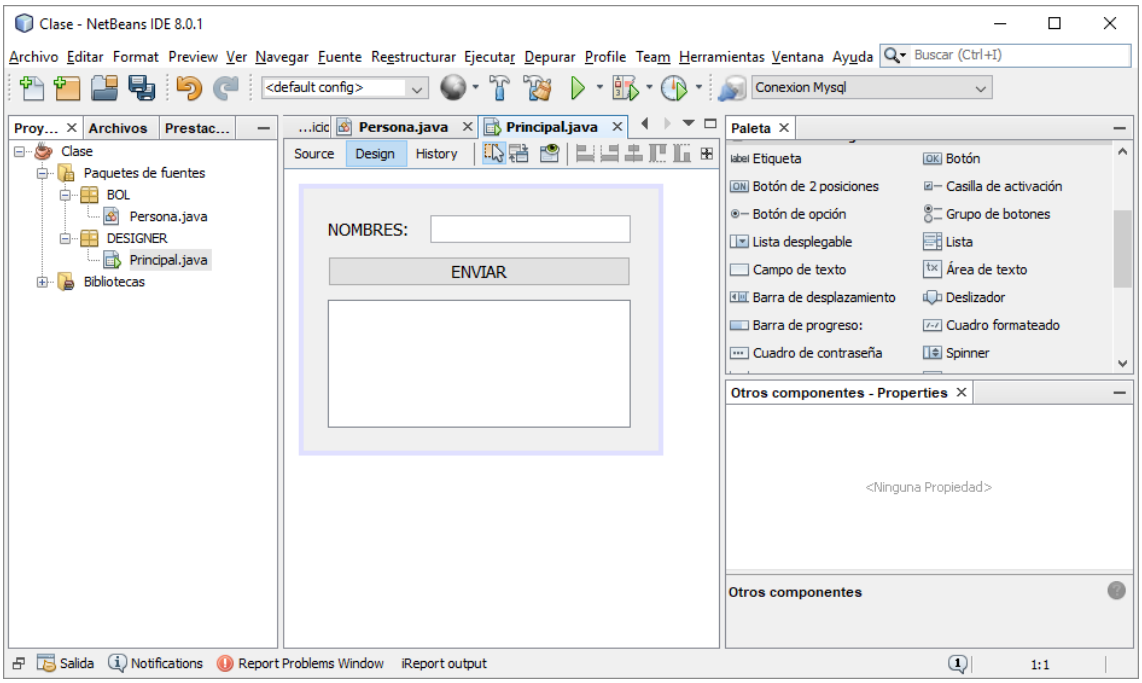


10. Dentro de la capa "Designer", creamos un JFrame que lo llamaremos "Principal"

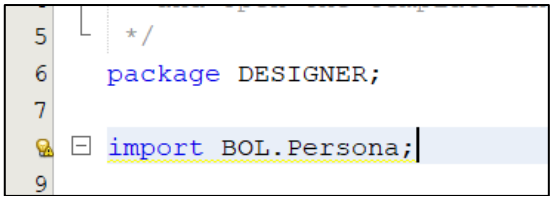




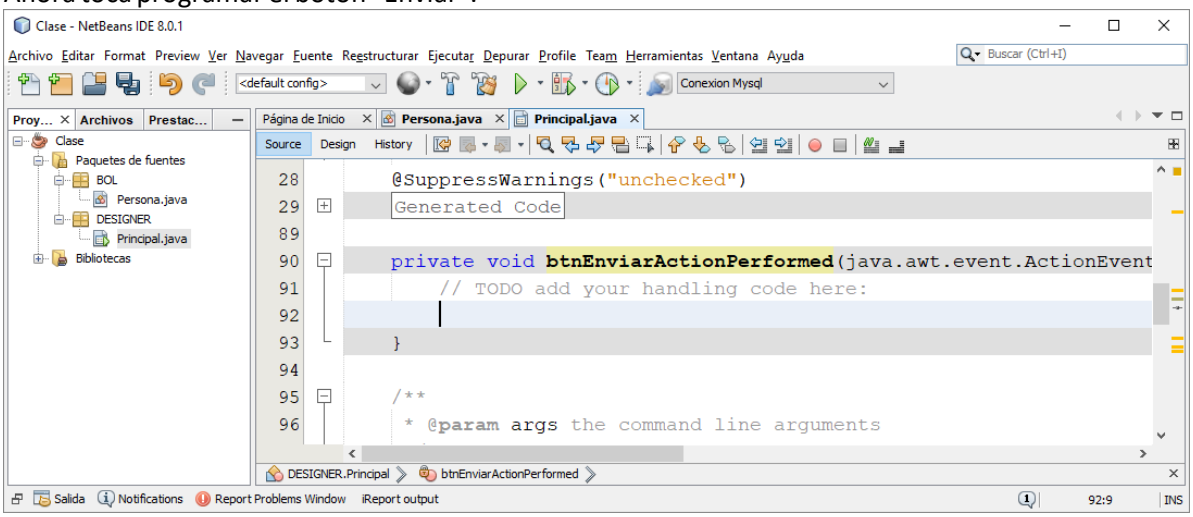
11. Nuestro Jframe "Principal", lo diseñamos de la siguiente manera.



12. Dentro el código de nuestro jframe, lo primero será importar la capa BOL para poder trabajar con nuestra clasePersona.



13. Ahora toca programar el botón “Enviar”.



14. Instanciamos la clase persona creando el objeto "objPer".

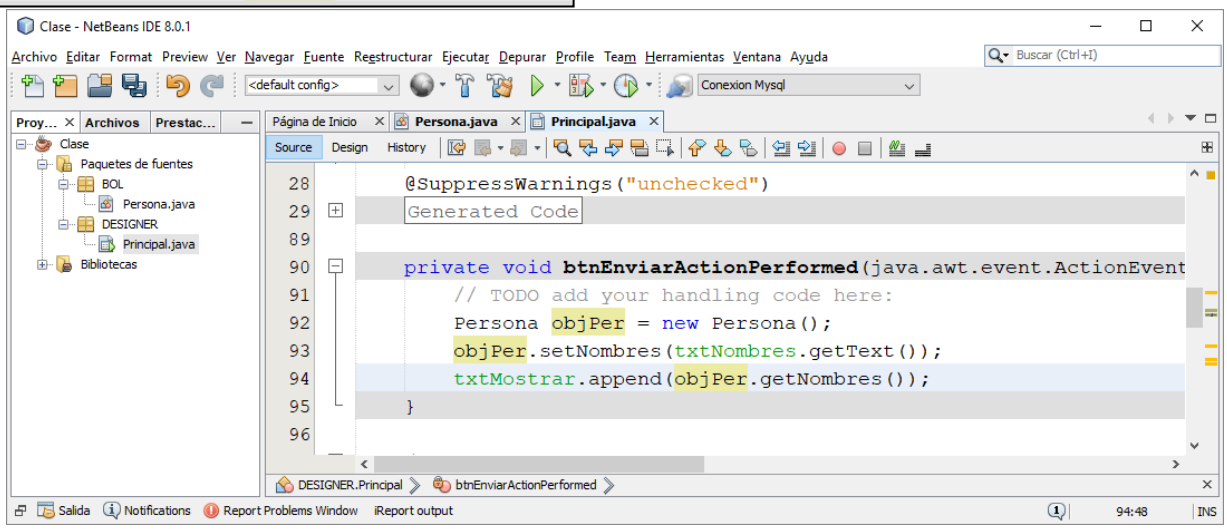
```
Persona objPer = new Persona();
```

15. Establecemos al "setNombres()" el valor que es ingresado mediante la caja de texto "txtNombres".

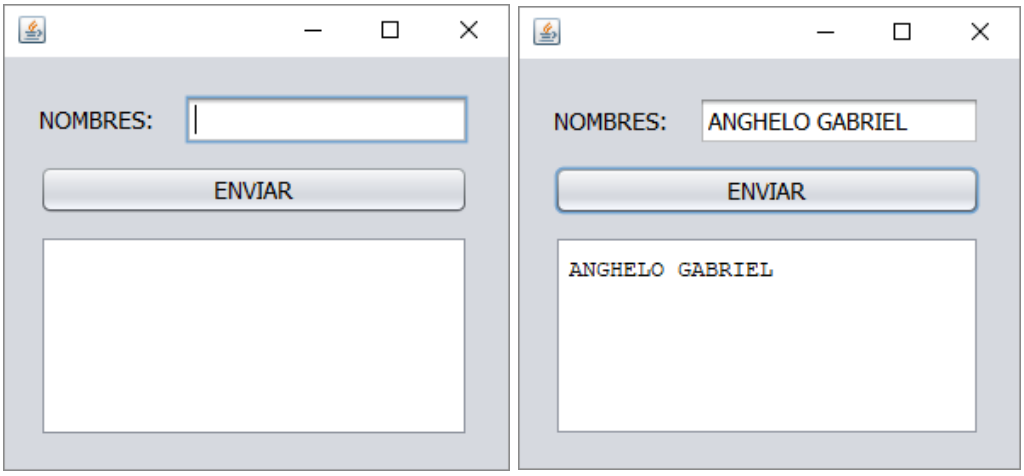
```
objPer.setNombres(txtNombres.getText());
```

16. Obtenemos mediante la funcion "getNombres()", el valor del campo nombres que fue cargado por "setNombres()", y lo añadimos a nuestra área de texto "txtMostrar"

```
txtMostrar.append(objPer.getNombres());
```



17. Ejecutamos y obtendremos lo siguiente.



PRACTICA

A. Agregar dentro de la clase "Persona", los siguientes campos:
   - ap_paterno(cadena)
   - ap_materno(cadena)
   - anio_nacimiento(entero)
   - dni(entero)
   - dirección(cadena)
   - sexo(caracter)

B. Crear lo correspondiente en el diseño del JFrame, contemplando lo siguiente:
   - Se ingresa el anio de nacimiento, pero se añade en el área de texto la edad.
   - Se ingresa el sexo (M/F), pero se añade en el área de texto "Masculino" o "Femenino".