	INSTITUCIÓN EDUCATIVA VILLA FLORA	CÓDIGO: ED-F-30	VERSIÓN 2
	Taller 1	FECHA: 23-02-2019	

Marque el tipo de taller: Complementario _____ Permiso _____ Desescolarización _____ Otro x
Asignatura: Media Técnica Grado: 11°1 Fecha: Marzo 17/2020

Docente: Gloria Cecilia Ríos Muñoz
Nombre y Apellidos de estudiante:

Propósito (indicador de desempeño):

- Conceptual: Interpreta la información técnica de diseño de construcción del software.

Pautas para la realización del taller:

- Seguir los pasos dados
- Entregar el cuaderno con la actividad propuesta
- Traer preguntas o dudas para socializar y resolver en clase

Ítems de Evaluación:

- Socialización Preguntas o dudas emitidas por el grupo. 30% Indicador conceptual
- Actividad propuesta en el cuaderno 70% Indicador conceptual

Actividades:

1. Conceptualización- **TEMA: Programación Orientada a Objetos POO,** debes hacer lectura de los siguientes conceptos dados.

   PROGRAMACIÓN ORIUENTADA A OBJETOS POO

   El concepto Programación Orientada a Objetos resulta un **poco confusa** para mucha gente al principio, cuando se entra en contacto con ella. Pero en este artículo se tratará de explicar con las palabras más sencillas posibles **los principales conceptos de la Programación Orientada a Objetos**, independientemente del lenguaje de programación que utilices corresponden a:

   CLASES, OBJETOS E INSTANCIAS

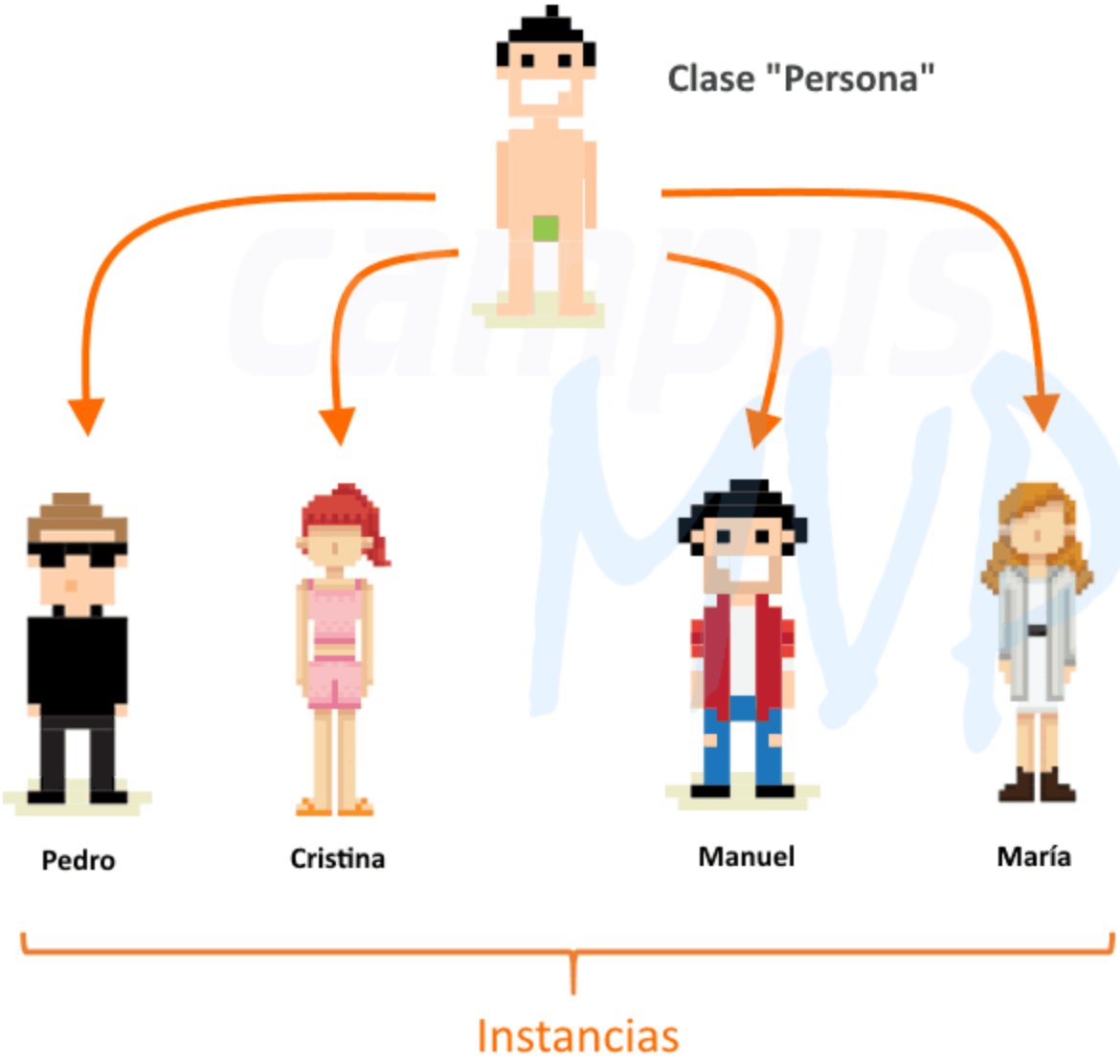
   El primer concepto y más importante de la POO es **la distinción entre clase y objeto**.

   Una clase es una plantilla. Define de manera genérica cómo van a ser los objetos de determinado tipo. Por ejemplo, en un juego, una clase para representar a personas puede llamarse Persona y tener una serie de **atributos** como Nombre, Apellidos o Edad (que normalmente son propiedades), y una serie de **comportamientos** que pueden tener, como Hablar(), Caminar() o Comer() y que se implementan como **métodos de la clase** (funciones).

   Una clase por sí sola no sirve de nada, pues no es más que un concepto, sin entidad real. Para poder utilizar una clase en un programa lo que hay que hacer es **instanciarla**. Instanciar una clase consiste en **crear un nuevo objeto concreto de la misma**. Es decir, **un objeto es ya una entidad concreta** que se crea a partir de la plantilla que es la clase. Este nuevo objeto tiene ya "existencia"

real, puesto que ocupa memoria y se puede utilizar en el programa. Así un objeto puede ser una persona que se llama Cristina López, de 37 años y que en nuestro programa podría hablar, caminar o comer, que son los comportamientos que están definidos en la clase.

Una imagen vale más que mil palabras:



De este modo, si tenemos que manejar personas podemos ir creándolas a medida que las necesitemos, y actuar sobre ellas individualmente. Cada una tiene sus propios datos y sus propias acciones.

La clase define de forma genérica cómo son las personas, y los objetos son personas concretas.

La POO es muy potente porque nos permite modelar de manera sencilla datos y comportamientos complejos del mundo real. Al poder manejar los datos y los comportamientos de cada objeto de manera independiente nos evita tener que mantener datos globales y coordinar todo eso. En su momento fue una verdadera revolución.

LOS CUATRO PILARES DE LA POO

Lo anterior por si solo es estupendo. Sin embargo, no es suficiente. Para poder manejar de manera eficiente las clases y los objetos que se generan con la Programación Orientada a Objetos son necesarios algunos principios que nos ayudarán a reducir la complejidad, ser más eficientes y evitar problemas. **Son los 4 pilares de la POO**. Todos los lenguajes orientados a objetos los implementan de una u otra manera, y es indispensable conocerlos bien.

Pilar 1: Encapsulación

El concepto de encapsulación es el más evidente de todos. Pero, precisamente por su sencillez, a veces pasa inadvertida.

La encapsulación es la característica de un lenguaje POO que **permite que todo lo referente a un objeto quede aislado dentro de éste**. Es decir, que todos los datos referentes a un objeto queden "encerrados" dentro de éste y sólo se puede acceder a ellos a través de los miembros que la clase proporcione (propiedades y métodos).

Por ejemplo, en el caso de las personas que estábamos viendo, toda la información sobre éstas (nombre, apellidos, edad... y cualquier otro dato interno que se utilice y que no necesariamente se ve desde el exterior del objeto) está circunscrito al ámbito de dicha persona.

Así, internamente tenemos un dato que es el nombre de la persona y accedemos a él a través de la propiedad pública Nombre que define la clase que representa a las personas. De este modo damos acceso sólo a lo que nos interese y del modo que nos interese. En un lenguaje tradicional tendríamos que montar alguna estructura global para almacenar esa información y luego acceder ordenadamente a ella, sin mezclar los datos de una persona con los de otra.

Gracias a la encapsulación, toda la información de un objeto está contenida dentro del propio objeto.

Pilar 2: Abstracción

Este concepto está muy relacionado con el anterior.

Como la propia palabra indica, el principio de abstracción lo que implica es que la clase debe **representar las características de la entidad hacia el mundo exterior, pero ocultando la complejidad** que llevan aparejada. O sea, nos abstrae de la complejidad que haya dentro dándonos una serie de atributos y comportamientos (propiedades y funciones) que podemos usar sin preocuparnos de qué pasa por dentro cuando lo hagamos.

Así, una clase (y por lo tanto todos los objetos que se crean a partir de ella) debe exponer para su uso solo lo que sea necesario. Cómo se haga "por dentro" es irrelevante para los programas que hagan uso de los objetos de esa clase.

En nuestro ejemplo de las personas en un juego, puede ser que tengamos un dato interno que llamamos energía y que nunca es accesible directamente desde fuera. Sin embargo, cada vez que la persona anda (o corre, si tuviésemos un método para ello) gasta energía y el valor de este dato disminuye. Y cuando la persona come, el valor sube en función de lo que haya comido.

Otro ejemplo incluso más claro podría ser la acción Hablar(). Ésta puede suponer que se genere una voz sintética a partir de un texto que se le indica como parámetro de la acción para lo cual quizá ocurran un montón de cosas: se llama a un componente para síntesis de voz que está en la nube, se lanza la síntesis de voz en el dispositivo local de audio y se anota en una base de datos la frase que se ha pronunciado para guardar un histórico entre otras cosas. Pero todo esto es indiferente para el programa que hace uso de esta funcionalidad. El programa simplemente tiene acceso a un objeto Cristina y llama a la función Hablar(). No tiene ni idea de toda la complejidad interna que puede

suponer. Si mañana cambiamos el modo de sintetizar la voz o cualquier otra acción interna, es indiferente para el programa que usa nuestros objetos de tipo Persona.

La abstracción está muy relacionada con la encapsulación, pero va un paso más allá pues no sólo controla el acceso a la información, sino también oculta la complejidad de los procesos que estemos implementando.

Pilar 3: Herencia

Desde el punto de vista de la genética, cuando una persona obtiene de sus padres ciertos rasgos (el color de los ojos o de la piel, una enfermedad genética, etc...) se dice que los hereda. Del mismo modo **en POO cuando una clase hereda de otra obtiene todos los rasgos que tuviese la primera**.

Dado que una clase es un patrón que define cómo es y cómo se comporta una cierta entidad, una clase que hereda de otra obtiene todos los rasgos de la primera y **añade otros nuevos** y además también **puede modificar algunos de los que ha heredado**.

A la clase de la que se hereda se le llama **clase base**, y a la clase que hereda de ésta se le llama **clase derivada**.

Así, en nuestro juego que involucra personas, podemos tener clases de personas más especializadas para representar personajes especiales del juego. Por ejemplo, podríamos definir clases como Pirata, Piloto o Estratega que heredan de la clase Persona. Todos los objetos de estas clases heredan las propiedades y los métodos de Persona, pero pueden particularizar algunos de ellos y además añadir cosas propias.

Por ejemplo, los objetos de la clase Pirata tienen un método nuevo que es Abordar () que en el juego sirve para asaltar un barco enemigo. Pero además presentan una propiedad que solo tienen los piratas llamada Sobrenombre, que es el nombre por el que se les conoce (un pirata puede ser de nombre Hızır y de apellido bin Yakup pero su sobrenombre es Barbaroja).

No solo eso. Lo bueno de la herencia es que podemos reutilizar todo lo que tuviésemos en la clase base. Supongamos que en nuestro juego, los piratas hablan de forma diferente a los demás. El método Hablar () se modifica para que le añada un ¡Arrrr! o un ¡Por todos los demonios! aleatoriamente a la frase y que así parezca más un pirata 😄. Para que el pirata hable no tendríamos que volver a hacer todo el código relacionado con hablar. Eso ya sabe cómo hacerlo por el mero hecho de ser una persona (por heredar de la clase Persona). Lo único que tendríamos que hacer es añadir esas interjecciones de pirata a la frase y luego delegar la síntesis de voz y todo lo demás a la clase base. Sería facilísimo y conseguiríamos consistencia entre todas las clases a la hora de particularizar la forma de hablar.

La herencia es una de las características más potentes de la POO ya que fomenta la reutilización del código permitiendo al mismo tiempo la particularización o especialización del mismo.

Pilar 4: Polimorfismo

La palabra polimorfismo viene del griego "polys" (muchos) y "morfo" (forma), y quiere decir "cualidad de tener muchas formas".

En POO, el concepto de polimorfismo se refiere al hecho de que **varios objetos de diferentes clases, pero con una base común, se pueden usar de manera indistinta**, sin tener que saber de qué clase exacta son para poder hacerlo.

Supongamos que en nuestro juego tenemos un montón de personajes que están juntos en un mismo escenario. Hay varios piratas, algunos estrategas y un montón de otros tipos de personas. En un

momento dado necesitamos que todos se pongan a hablar. Cada uno lo hace de una forma diferente, ya que son tipos de personajes distintos. Sería algo bastante tedioso tener que localizar primero a los de un tipo y hacerlos hablar, lo luego a los de otro y así sucesivamente. La idea es que puedas tratarlos a todos como personas, independientemente del tipo específico de persona que sea y simplemente decirles que hablen.

Al derivar todos de la clase Persona todos pueden hablar, y al llamar al método Hablar () de cada uno de ellos se utilizará el proceso adecuado según el tipo (los piratas meterán sus expresiones adicionales que hemos visto, los pilotos dirán "Entrando en pista" o lo que sea, y los estrategas añadirán a todo "Déjame que lo piense bien"). Todo esto de manera transparente para el programador. Esto es el polimorfismo.

De hecho, el polimorfismo puede ser más complicado que eso ya que se puede dar también mediante la sobrecarga de métodos y, sobre todo, a través del uso de interfaces, pero el concepto es el que acabo de explicar.

El polimorfismo nos permite utilizar a los objetos de manera genérica, aunque internamente se comporten según su variedad específica.

Gracias a estos cuatro principios que cumplen todos los lenguajes orientados a objetos se facilita mucho la programación de ciertos tipos de problemas, se minimizan errores, se escribe código más rápido y se puede mantener más fácilmente cuando haya modificaciones en el futuro.

Cada lenguaje tiene su sintaxis específica para crear objetos y expresar los cuatro pilares, pero estos conocimientos genéricos te valdrán para cualquiera de ellos.

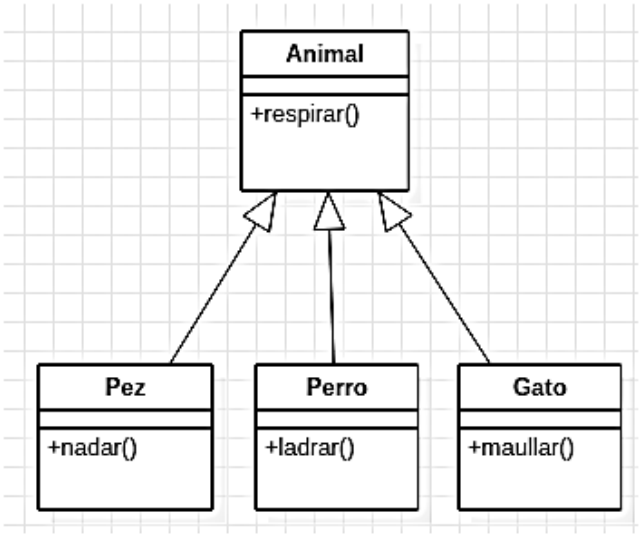
2. Puedes ingresar al enlace compartido en los recursos: http://www.profmatiasgarcia.com.ar/uploads/tutoriales/2PilaresOrientacionAObjetos.pdf o buscar otros referentes en bases de datos confiables que te permitan afianzar los conceptos.
3. Contestar el siguiente cuestionario en tu cuaderno:

- De acuerdo a los conceptos dados y consultados, analiza la siguiente ilustración y emite 4 ejemplos de cada uno de los pilares que representan la POO. Regístralos en el Cuaderno.

- Responde las siguientes preguntas de acuerdo a la lectura de los conceptos anteriores en tu cuaderno:

a) ¿Qué función tiene los cuatro pilares de la POO?
b) ¿Qué es Encapsulamiento?
c) Si tenemos una clase llamada Vehículo, cuál cree que serán los métodos?
d) ¿Cuál de los pilares permite que una clase (clase hija o subclase) reciba los atributos y métodos de otra clase (clase padre o superclase)?
e) Si queremos hacer una representación de la abstracción con las “Aves”, ¿cuál sería el objeto?, ¿Cuáles serías sus características?, ¿Cuáles serían sus funcionalidades asociadas?
f) ¿Cuáles son las características del pilar encapsulación en la POO?
g) ¿Qué entiendes por modularidad?
h) Analiza la siguiente imagen y deduce a cuál de los pilares pertenece según tu interpretación:

i) De acuerdo a la siguiente imagen elabora la herencia de una clase llamada Transporte



Ejemplo de herencia

4. Repasa la actividad y prepara preguntas o dudas para socialización, en la próxima clase.
5. Presentar el cuaderno con la actividad completa para la próxima clase.

Recursos:

- Cuaderno físico
- Edmodo se actividad en el momento indicado
- Internet
- Pc
- Enlace: http://www.profmatiasgarcia.com.ar/uploads/tutoriales/2PilaresOrientacionAObjetos.pdf

	INSTITUCIÓN EDUCATIVA VILLA FLORA	CÓDIGO: ED-F-30	VERSIÓN 2
	Taller 2	FECHA: 23-02-2019	

Marque el tipo de taller: Complementario _____ Permiso _____ Desescolarización _____ Otro x
Asignatura: Media Técnica Grado: 11°1 Fecha: Marzo 187/2020

Docente: Gloria Cecilia Ríos Muñoz
Nombre y Apellidos de estudiante:
_____

Propósito (indicador de desempeño):

- Procedimental: Codificar el Software utilizando el lenguaje de programación y la plataforma seleccionada

Pautas para la realización del taller:

- Seguir los pasos dados
- Crear dos carpetas en Java, llamadas exploración y la otra Aplicación
- Trabajo individual
- Entregar 2 archivos de forma digital los cuales se deben comprimir y publicar en Edmodo de forma individual
- Prepara preguntas o dudas sobre la exploración y aplicación

Ítems de Evaluación:

- Ejercicio de Exploración del lenguaje de programación JAVA. 30% Indicador Procedimental
- Ejercicio de Aplicación del lenguaje de programación JAVA. 60% Indicador Procedimental
- Evaluación con Socialización en clase – preguntas. 10% Indicador Procedimental

Actividades:

1. Leer los siguientes conceptos sobre el Tema: Introducción a JAVA PARTE I

Es importante tener muy claro los siguientes conceptos: (Puedes tomar nota, hacer copiar y pegarlos en tu cuaderno), lo más importante es que asimiles dichos conceptos. Pero debes seguir el paso a paso para su comprensión.

¿Qué es una clase?

Las clases en Java son básicamente una plantilla que sirve para crear un objeto. Si imaginásemos las clases en el mundo en el que vivimos, podríamos decir que la clase «**persona**» es una plantilla sobre cómo debe ser un ser humano. Todos y cada uno de nosotros, los seres humanos, somos objetos de la clase «**persona**», ya que todos somos personas. La clase **«persona»** contiene la definición de un ser humano, mientras que cada ser humano es una instancia u objeto de dicha clase.

¿Qué es un main()método?

En Java, cada aplicación debe contener un main()método, que es el punto de entrada para la aplicación. Todos los demás métodos se invocan desde el main()método.

¿Qué es una Línea de impresión?

- System.out.println() puede imprimir en la consola:
- System es una clase de la biblioteca principal proporcionada por Java
- out es un objeto que controla la salida
- println() es un método asociado con ese objeto que recibe un único argumento

¿Qué son los Comentarios?

Los comentarios son fragmentos de texto que el compilador ignora. Se utilizan para aumentar la legibilidad de un programa. Los comentarios de una sola línea se crean mediante el uso //. Los comentarios de varias líneas se crean comenzando /*y terminando con */.

¿Que son los Espacio en blanco?

Se ignora el espacio en blanco, incluidos los espacios y las nuevas líneas, entre las declaraciones.

Compilando Java

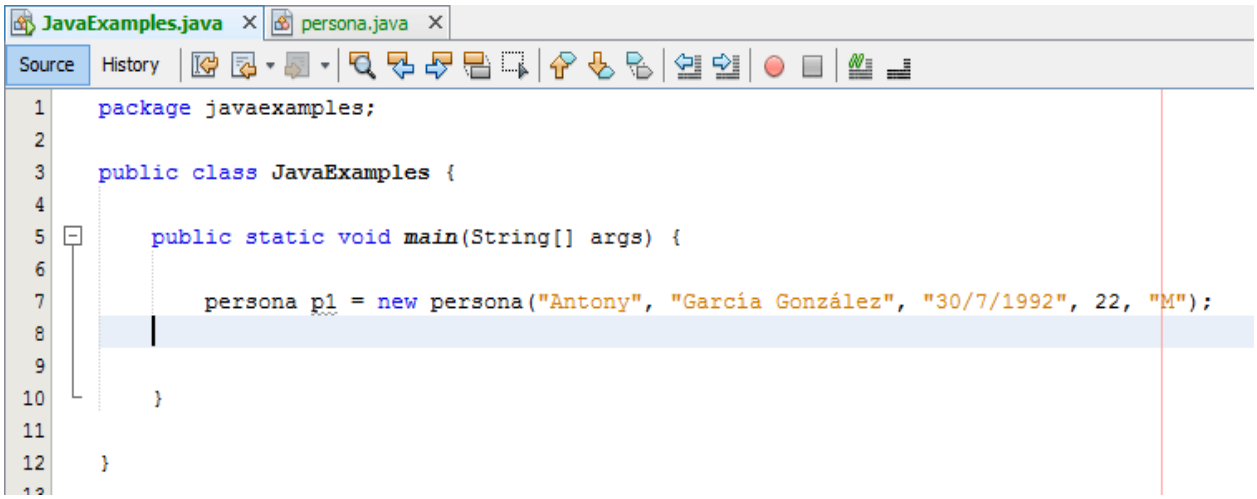
En Java, cuando compilamos un programa, cada clase individual se convierte en un archivo **.class**, que se conoce como código de bytes. La JVM (máquina virtual Java) se usa para ejecutar el código de bytes.

¿Qué son Declaraciones?

En Java, una instrucción es una línea de código que ejecuta una tarea y termina con un ;.

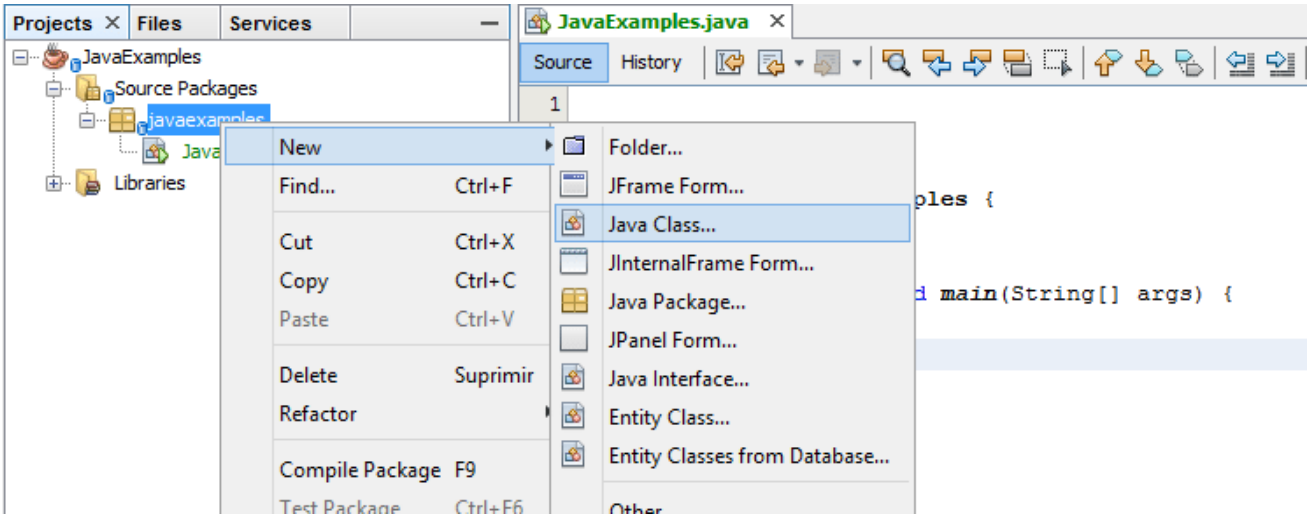
2. Deben observar los enlaces suministrados en los recursos, donde se comparte material de apoyo como: videos, tutoriales y enlaces para descargar el lenguaje de programación y poder efectuar la exploración y aplicación de JAVA.
3. Para poder comprender esto debemos primero explorar, ver con ejemplos. Vamos a crear un proyecto en Netbeans al que voy a llamar JavaExamples. Debes seguir el Paso a paso.

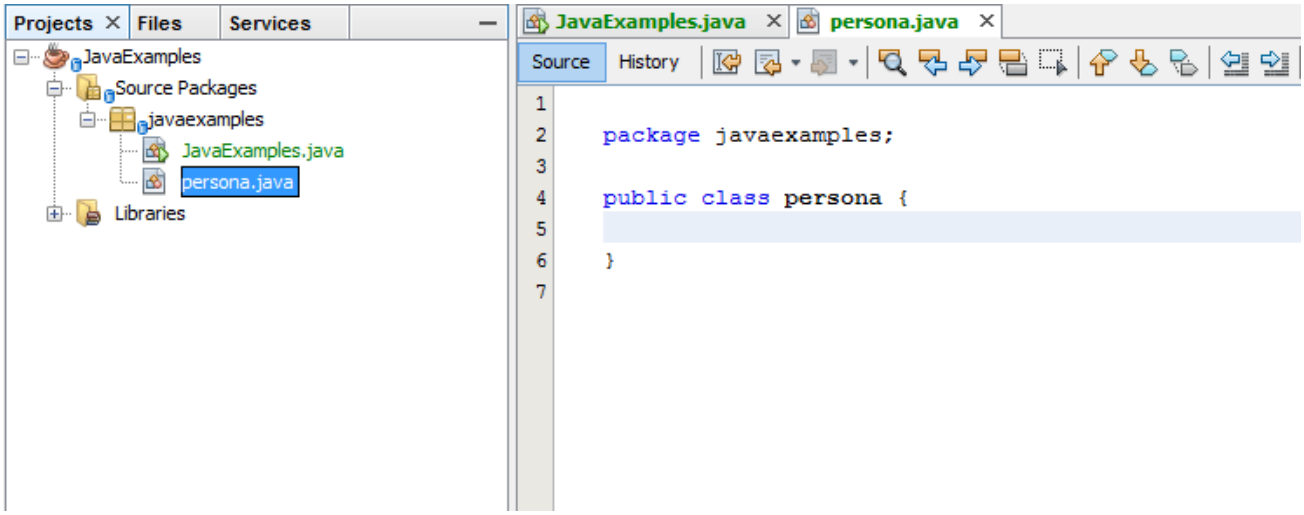
Recuerda crear una carpeta donde debe quedar guardado cada ejemplo del paso a paso. (En la parte inferior encuentras videos de cómo y donde puedes descargar el lenguaje de programación Netbeans,



Vemos que se crea un archivo llamado JavaExamples.java (Java File Class). Esta es la clase principal, la que contiene el método public static void main. ¨Al ejecutarse un programa, lo que se

encuentre establecido en el void main será lo primero en ejecutarse. La clase JavaExamples es un ejemplo de una clase, pero en ella no quedan claros los conceptos que me gustaría transmitirles, así que vamos a crear una clase a la que llamaremos «persona».





Ahora vamos a empezar a construir la clase «persona». Necesitamos establecer las **propiedades de la clase persona**. Como personas, ¿qué propiedades poseemos?

Podríamos hablar de un nombre(s), apellido(s), una fecha en la que nacimos, nuestra edad y género. Hay muchas otras propiedades que podríamos mencionar, pero por ahora solamente tomaremos en cuenta las que hemos listado. Son simplemente características que posee todo ser humano, al menos en la mayoría de los casos.

Para establecer estas propiedades dentro de la clase creamos variables, cuidándonos de asignarle a cada propiedad el tipo de dato que le corresponde. Dicho esto, declaramos las propiedades de la clase persona:

```
1 private String name;
2     private String lastName;
3     private String birthDate;
4     private int age;
5     private String gender;
```

Al establecer las propiedades como **private** estamos asegurando que dentro de la clase persona se pueda acceder y manipular estas propiedades, pero no así desde otra clase, a menos que el usuario así lo decida y se lleve a cabo a través de métodos públicos. Esto lo veremos un poco más adelante.

******Observación: Es necesario resaltar que, en la medida de lo posible, se debe utilizar nombres en inglés para las variables.*** *Se puede usar el idioma que sea y la estructura que nos plazca, pero se recomienda nombres en inglés y que las variables empiecen con minúscula. Si se trata de nombres compuestos de dos o más palabras se utilizan la primera palabra en minúscula y seguido la segunda palabra en mayúscula. Ejemplo: para llamar a una variable como* ***Last Name****, usamos* ***lastName****.*

Una vez establecidas las propiedades de la clase creamos **un constuctor**. **El contructor** es un método con el nombre de la clase donde se establecen los parámetros y las acciones a ejecutar cuando sea que se cree un nuevo objeto o instancia (objeto e instancia es básicamente lo mismo) de la clase persona. En Java las llamadas «funciones» de otros lenguajes se conocen como «métodos». Nuestro constructor sería:

```
1  public persona(String nombre, String apellido, String fecha, int edad, String genero) {
2          name = nombre;
3          lastName = apellido;
4          birthDate = fecha;
5          age = edad;
6          gender = genero;
7      }
```

Cuando vayamos a construir un objeto de la clase **persona** nos veremos obligados a establecer una serie de parámetros iniciales. Al crearse la instancia, la primera tarea que llevará a cabo la clase será asignarle los parámetros del constructor a las propiedades de la clase. Estas propiedades, como ya mencionamos, tienen un acceso privado dentro de la clase y no pueden ser modificadas desde el exterior. Sin embargo, podemos otorgarle acceso restringido desde las clases externas a las

propiedades de nuestra clase «**persona**«. Para ello establecemos los llamados **getters** y **setters** que son métodos para obtener o modificar propiedades de una clase.

Un ejemplo sería el setter y el getter para la propiedad Nombre.

```
1   public void setName(String nombre) {
2         name = nombre;
3    }
4
5    public String getName() {
6         return name;
7    }
```

Al invocar el setter ***setName(String nombre)*** estamos cambiando la propiedad name por el parámetro que l introduzcamos, o sea nombre. Este método es del tipo void, por lo que no devuelve ningún valor. Simplemente cambia la propiedad **name** y nada más.

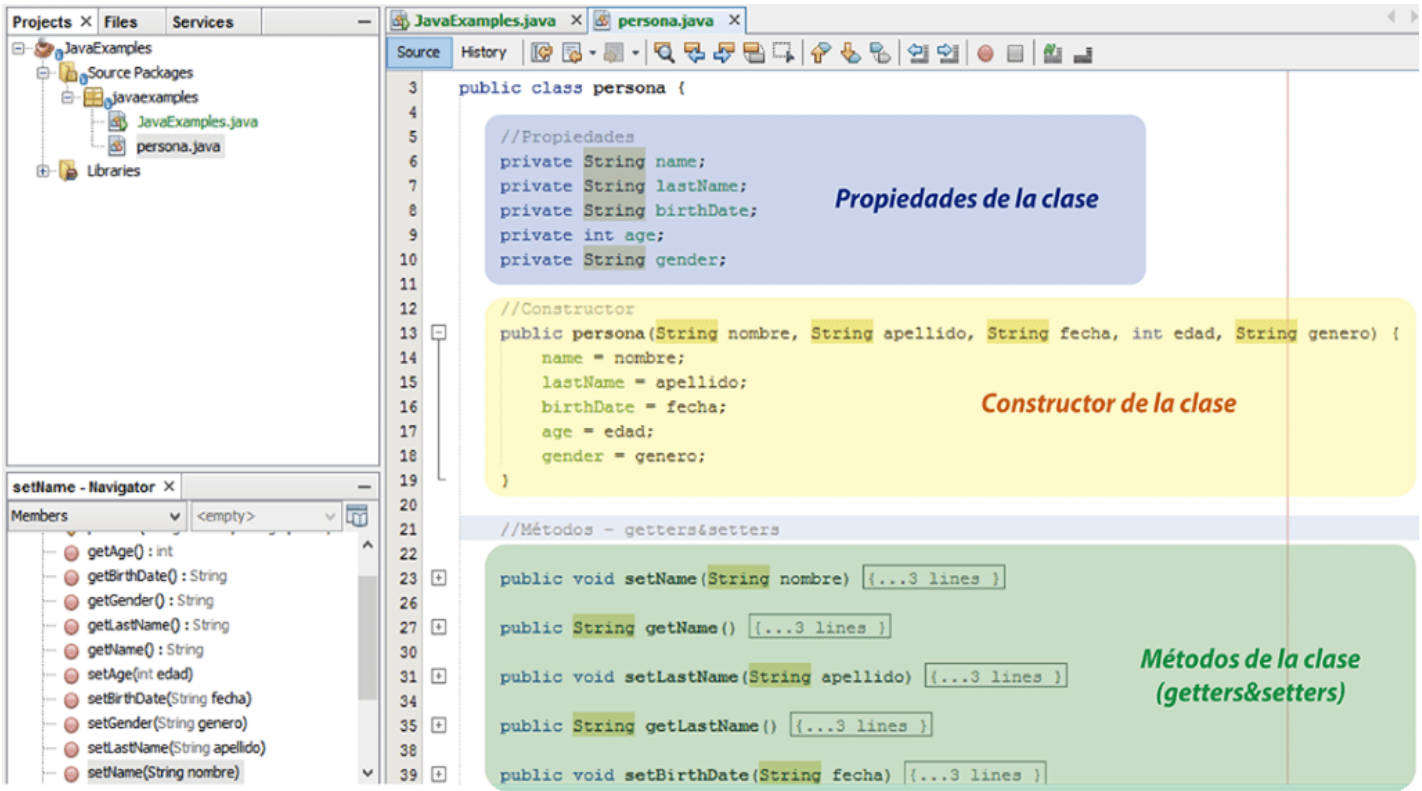
Al invocar el getter **getName()** vamos a «llamar» a la propiedad name y obtendremos su valor. El método es tipo String, o sea que devuelve una cadena de caracteres con el valor que tenga la propiedad name, de ahí que se use el comando **return name**.

Ahora establecemos los getters y setters para todas las propiedades. Nuestra clase persona quedaría así:

```
1  public class persona {
2
3      //Propiedades
4      private String name;
5      private String lastName;
6      private String birthDate;
7      private int age;
8      private String gender;
9
10     //Constructor
11     public persona(String nombre, String apellido, String fecha, int edad, String genero) {
12         name = nombre;
13         lastName = apellido;
14         birthDate = fecha;
15         age = edad;
```

```
16        gender = genero;
17    }
18
19    //Métodos - getters&setters
20
21    public void setName(String nombre) {
22        name = nombre;
23    }
24
25    public String getName() {
26        return name;
27    }
28
29    public void setLastName(String apellido) {
30        lastName = apellido;
31    }
32
33    public String getLastName() {
34        return lastName;
35    }
36
37    public void setBirthDate(String fecha) {
38        birthDate = fecha;
39    }
40
41    public String getBirthDate() {
42        return birthDate;
43    }
44
45    public void setAge(int edad) {
46        age = edad;
47    }
48
49    public int getAge() {
50        return age;
51    }
52
53    public void setGender(String genero) {
54        gender = genero;
55    }
56
57    public String getGender() {
58        return gender;
59    }
60
61 }
```

Entonces la estructura interna de nuestra clase «persona» debe lucir así:



Esta lista nuestra clase «personas». Hemos creado una plantilla para «crear» personas. Pero, ¿cómo es esto?. Nos vamos a nuestra clase principal y creamos una instancia de la clase persona.

1 persona p1 = new persona("Antony", "García González", "30/7/1992", 22, "M");

Esta persona nueva que he creado con el nombre de p1 posee como nombre «Antony», apellidos «García González», fecha de nacimiento «30/7/1992», edad 22 años y sexo masculino.

El objeto p1 me representa a mí. Yo soy Antony García González, pero dentro de Java soy un objeto de nombre p1. Este objeto lo creamos siguiendo la siguiente sintaxis:

Nombre_de_la_clase nombre_de_objeto = new **Nombre_de_la_clase** (parámetros del constructor).

JavaExamples.java | persona.java

Source | History

```
1  package javaexamples;
2
3  public class JavaExamples {
4
5      public static void main(String[] args) {
6
7          persona p1 = new persona("Antony", "García González", "30/7/1992", 22, "M");
8
9
10     }
11
12 }
13
```

Las clases en Java siempre se instancian de la misma manera. Instanciar es crear un objeto de una clase. Entonces este objeto que creamos, dentro de nuestra aplicación se llama **p1** pero para el usuario del software tiene mi nombre, Antony.

Del objeto **p1** podemos «obtener» sus propiedades o bien, «modificar» las mismas. Si escribimos en nuestro código «p1.» nos aparecerá lo siguiente:

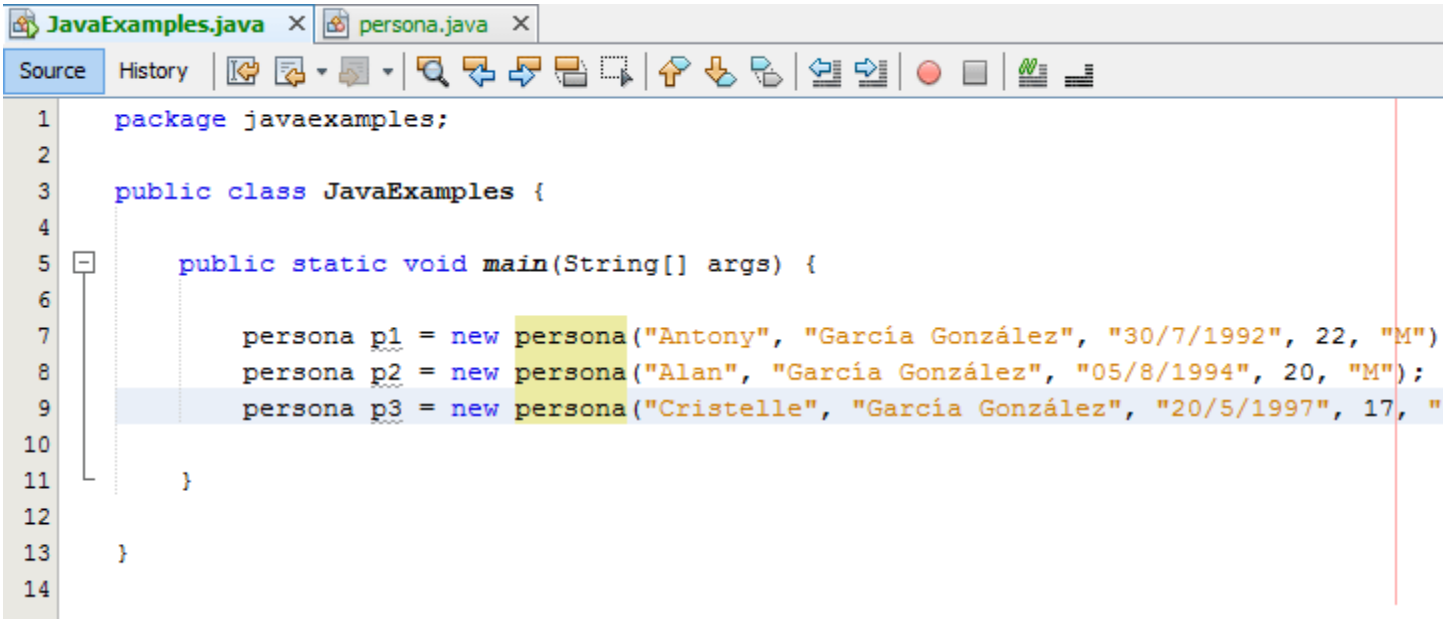
```
p1.
```

equals(Object o)	boolean
getAge()	int
getBirthDate()	String
getClass()	Class<?>
getGender()	String
getLastName()	String
getName()	String
hashCode()	int
notify()	void
notifyAll()	void
setAge(int edad)	void
setBirthDate(String fecha)	void
setGender(String genero)	void
setLastName(String apellido)	void
setName(String nombre)	void
toString()	String
wait()	void

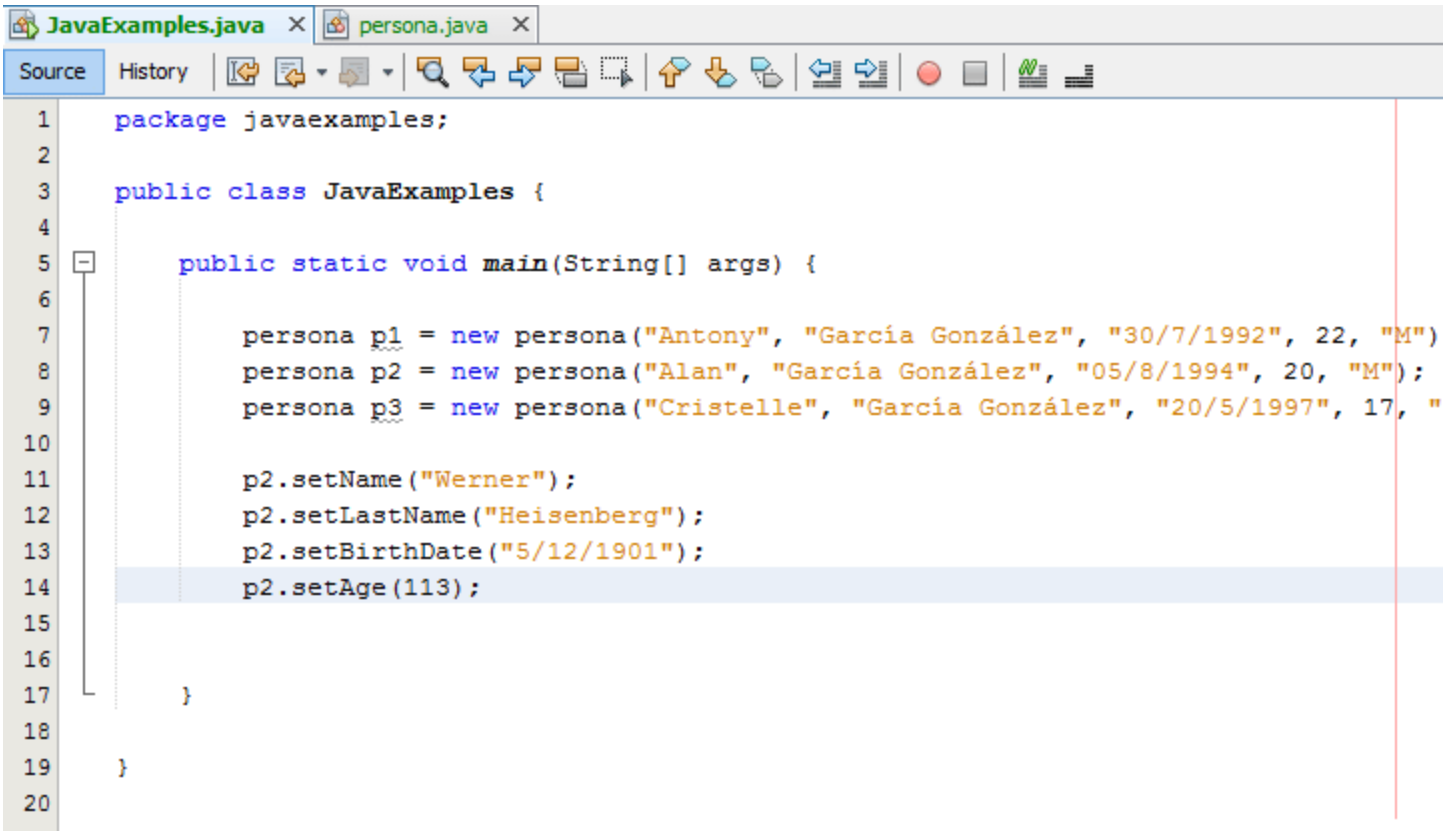
Vemos que el asistente de Netbeans nos presenta todos los métodos disponibles para el objeto p1. Esa es la ventaja de las clases en Java que un solo objeto contiene una gran cantidad de métodos en su interior para llevar a cabo distintos procedimientos resumidos en una sola línea de código. Por ahora hemos trabajado con sentencias cortas, solo establecer propiedades u obtener los valores de las mismas. Pero bien podríamos agregar un método que haga algo mucho más complicado y extenso e invocarlo luego de haber creado un objeto de la clase en cuestión.

Hay algunos métodos que no establecimos, como equals, hashCode, etc… Estos son métodos heredados de la clase Object. El concepto de herencia lo estudiaremos en otros aportes. Por ahora basta con saber que tenemos a nuestra disposición los getters y setters que establecimos al construir nuestra clase «persona».

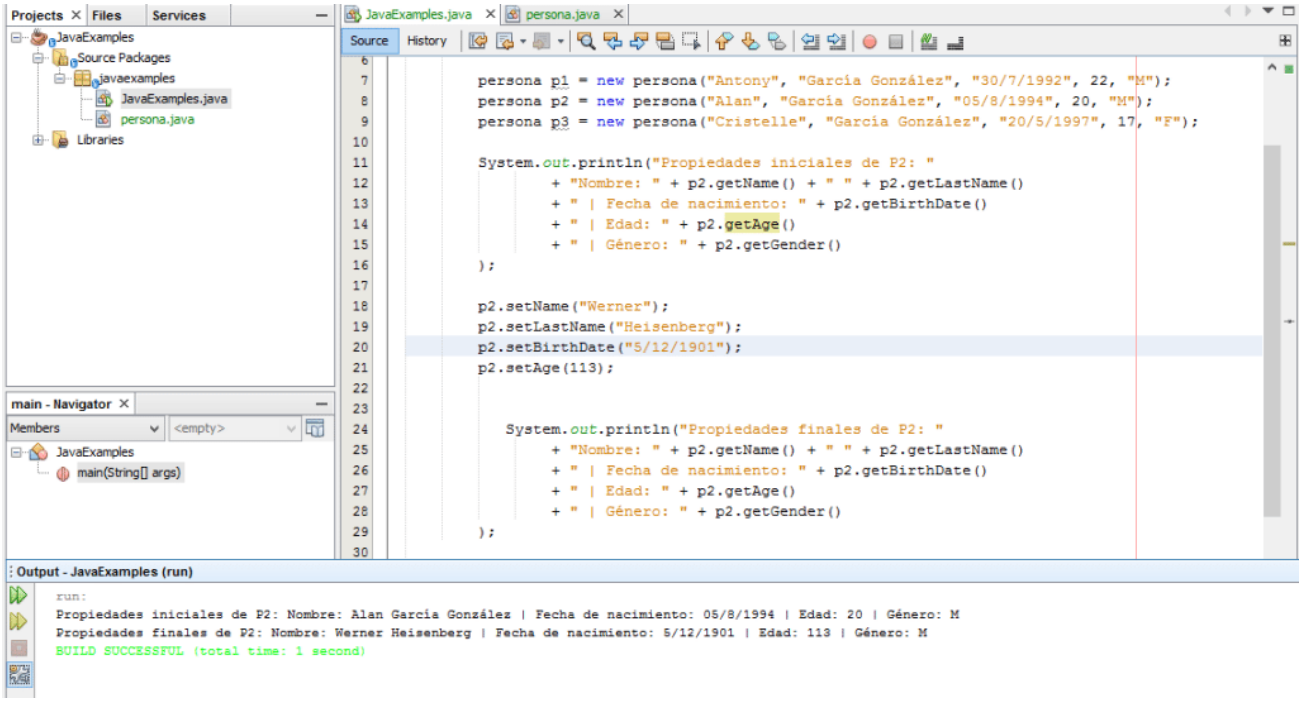
Podemos construir cuantos objetos queramos.



Podemos repentinamente cambiar algún nombre de algún objeto.



Podemos probar los getters imprimiendo en consola la información de **p2**.



Al usar los getters simplemente obtenemos cadenas de caracteres con los valores de las propiedades establecidas. Estas cadenas las imprimimos en la consola con el System.out.println y en la parte inferior de la ventana vemos los resultados. En la primera impresión el nombre y demás propiedades son los que establecimos en la creación del objeto p2. Luego con los setters cambiamos algunas propiedades y al volver a imprimir vemos que cambió la información.

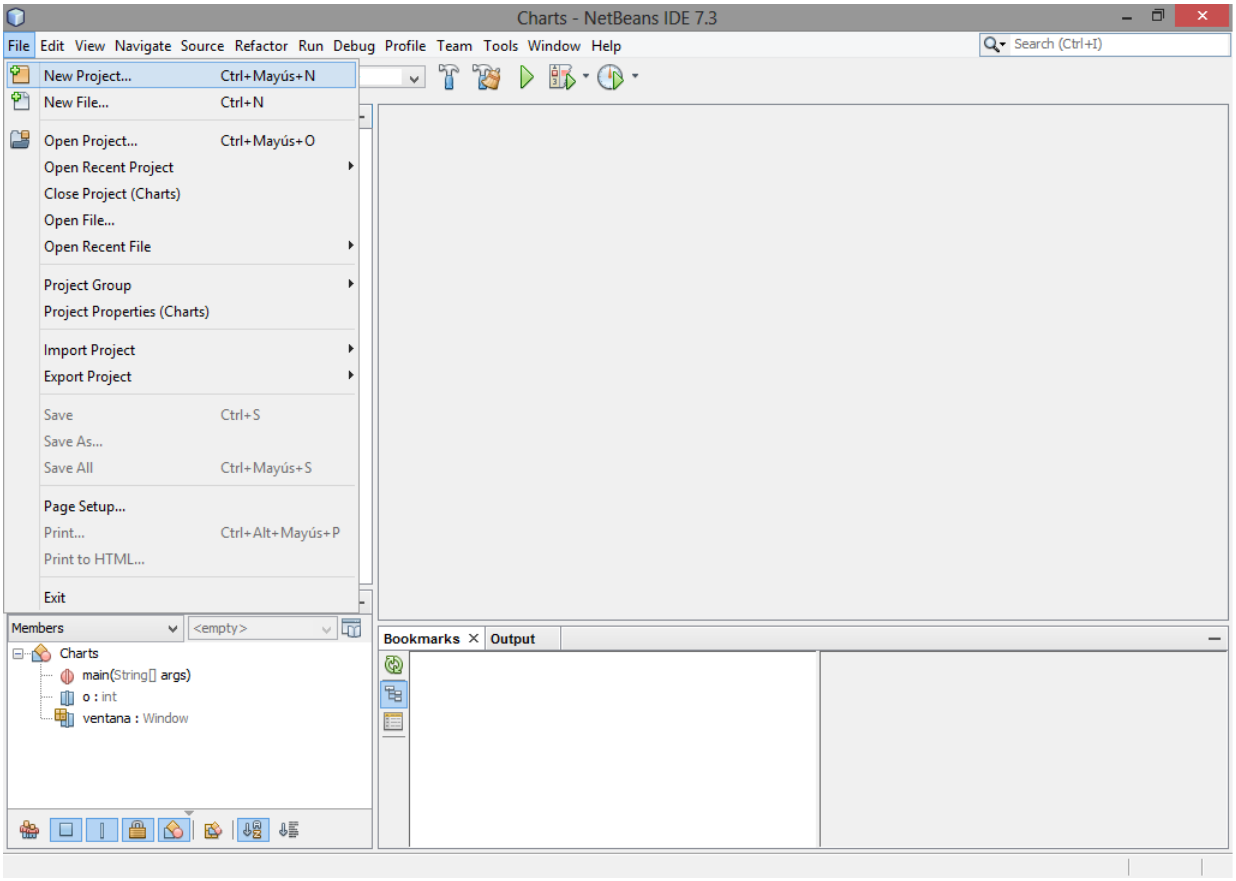
4. Ahora después de la exploración vamos a crear nuestro primer programa en Java. Es el momento de Apliación.

AHORA VAMOS A CREAR NUESTRO PRIMER PROGRAMA

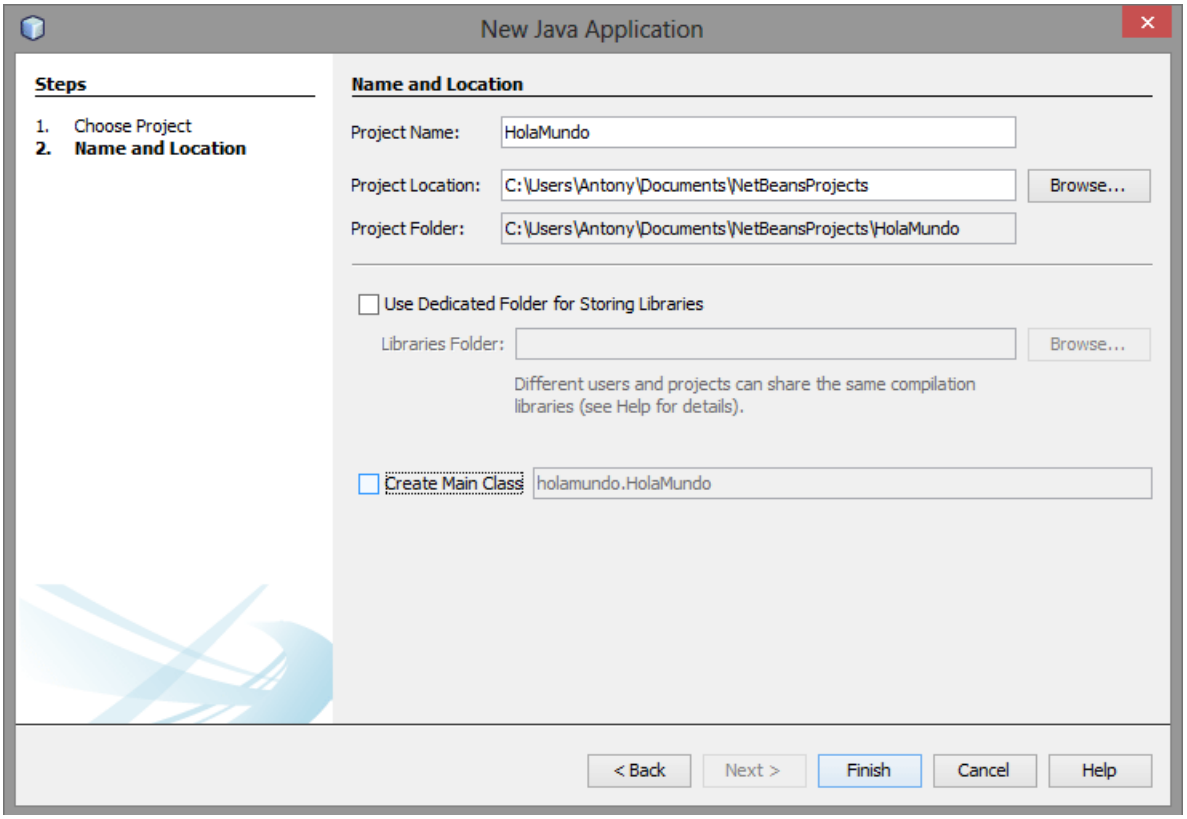
EN JAVA CON NETBEANS

En este post aprenderemos como crear una sencilla aplicación «Hola Mundo» en Java. Sigue paso a paso el ejercicio:

Para ello abriremos nuestro NetBeans IDE y nos dirigimos al menú File (o Archivo) en la opción New Project.

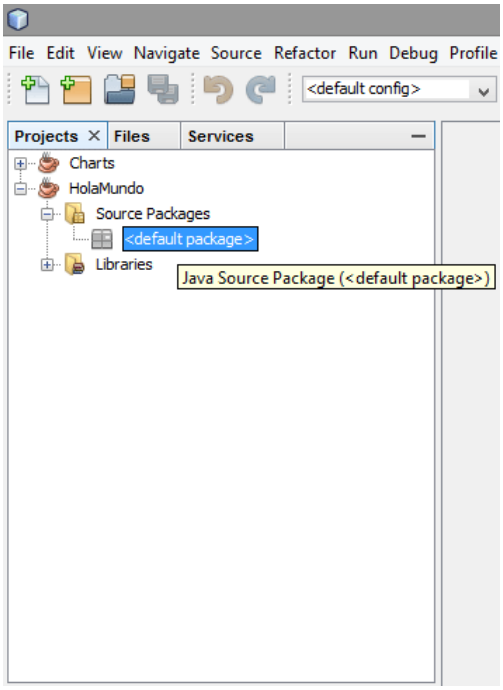


Ahora le asignamos un nombre a nuestro proyecto. Yo le he llamado HolaMundo. Debemos quitarle el ganchito a la casilla «Create Main Class» tal y como se muestra en la siguiente imagen.

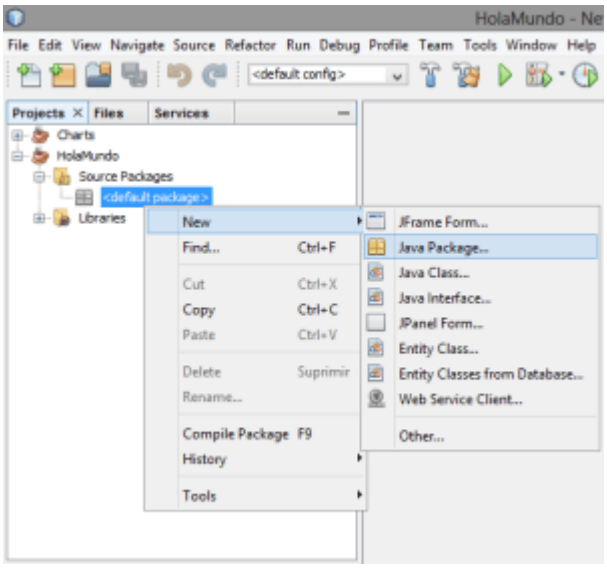


Luego de esto le damos clic en Finish. Con esto habremos creado nuestro proyecto. Ahora nos fijamos en la sección de la izquierda, nos aparecerá un árbol desplegable con nuestro proyecto. Desplegamos el contenido y encontramos **Source Packages** y **Library**.

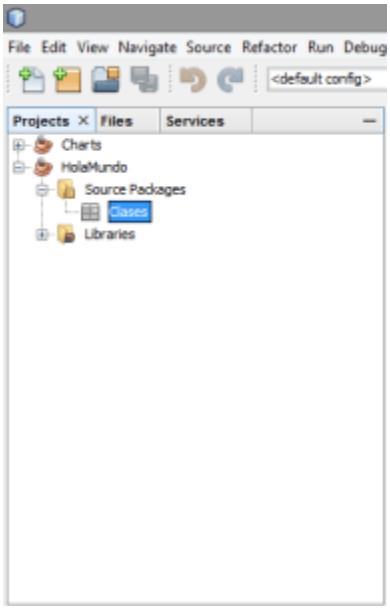
Ahora, dentro de Source Packages encontraremos **<default package>.**



Le damos clic derecho a **<default package>,** luego en «New» y buscamos la opción **Java Package**.

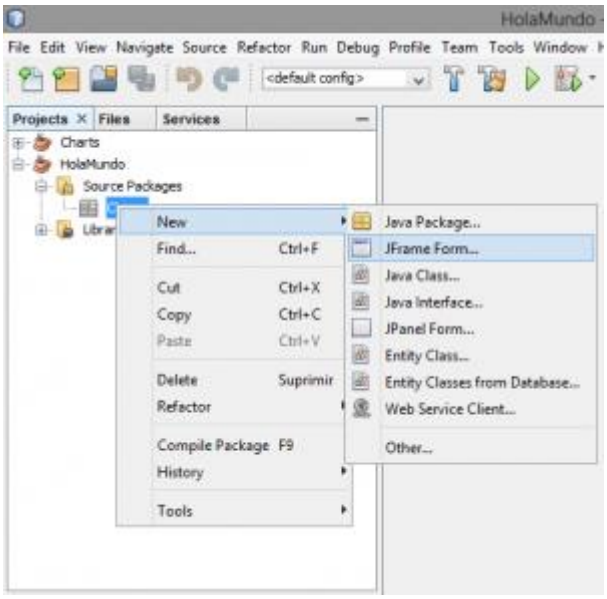


Creamos un nuevo paquete llamado Clases.

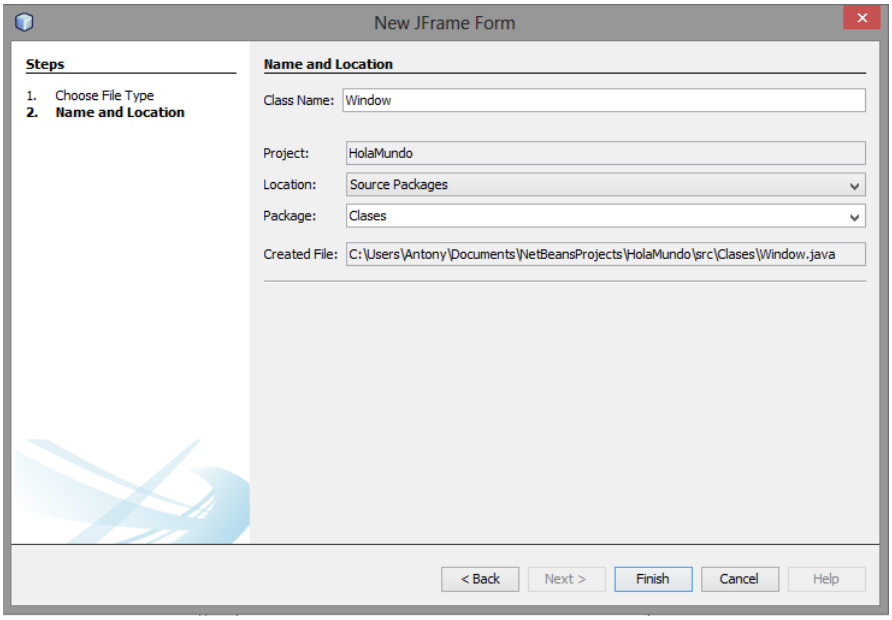


Le damos clic en «Finish».

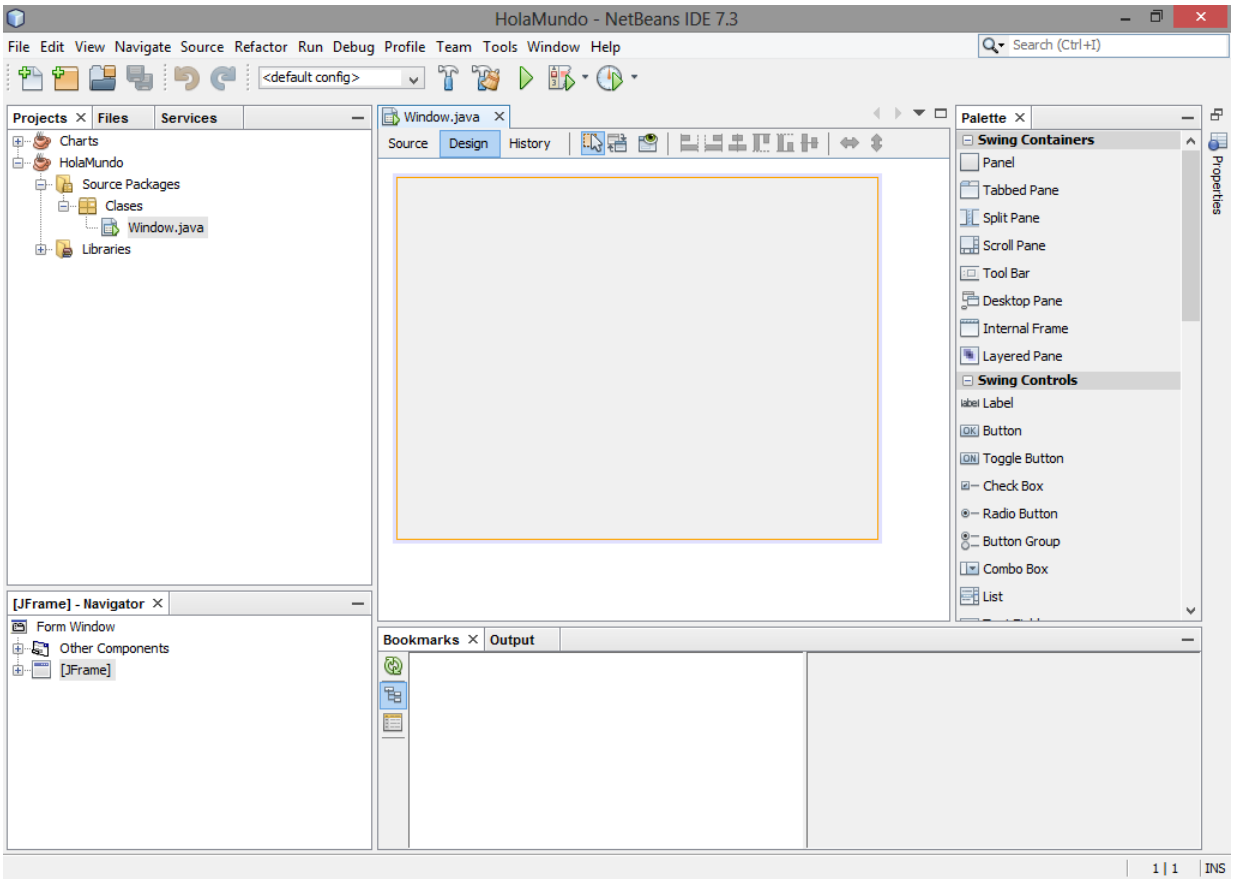
Ahora, de nuevo en el árbol desplegable. En el nuevo paquete llamado **Clases** damos clic derecho, «New», «JFrameForm».



Creamos un nuevo JForm llamado **«Window».**



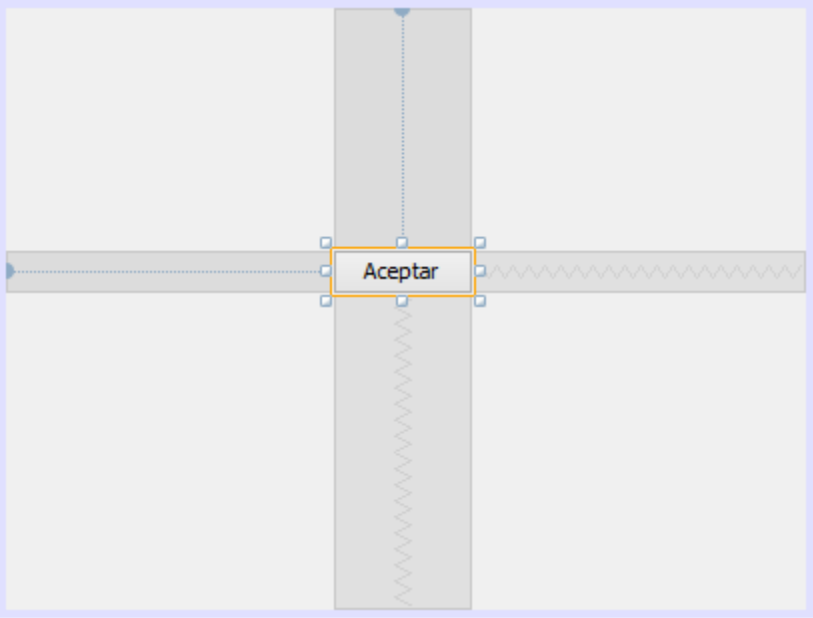
Le damos clic en «Finish» y nos aparecerá un cuadro en blanco. Este cuadro se conoce como JFrame y es el lienzo para iniciar la creación de un programa.



En la parte de la derecha encontraremos la **Paleta** o **Palette.**

Desplegamos Swing Controls y buscamos la opción que dice «Button» o «Boton».

Lo seleccionamos y lo arrastramos al JFrame. Con esto habremos agregado un botón a nuestro programa.

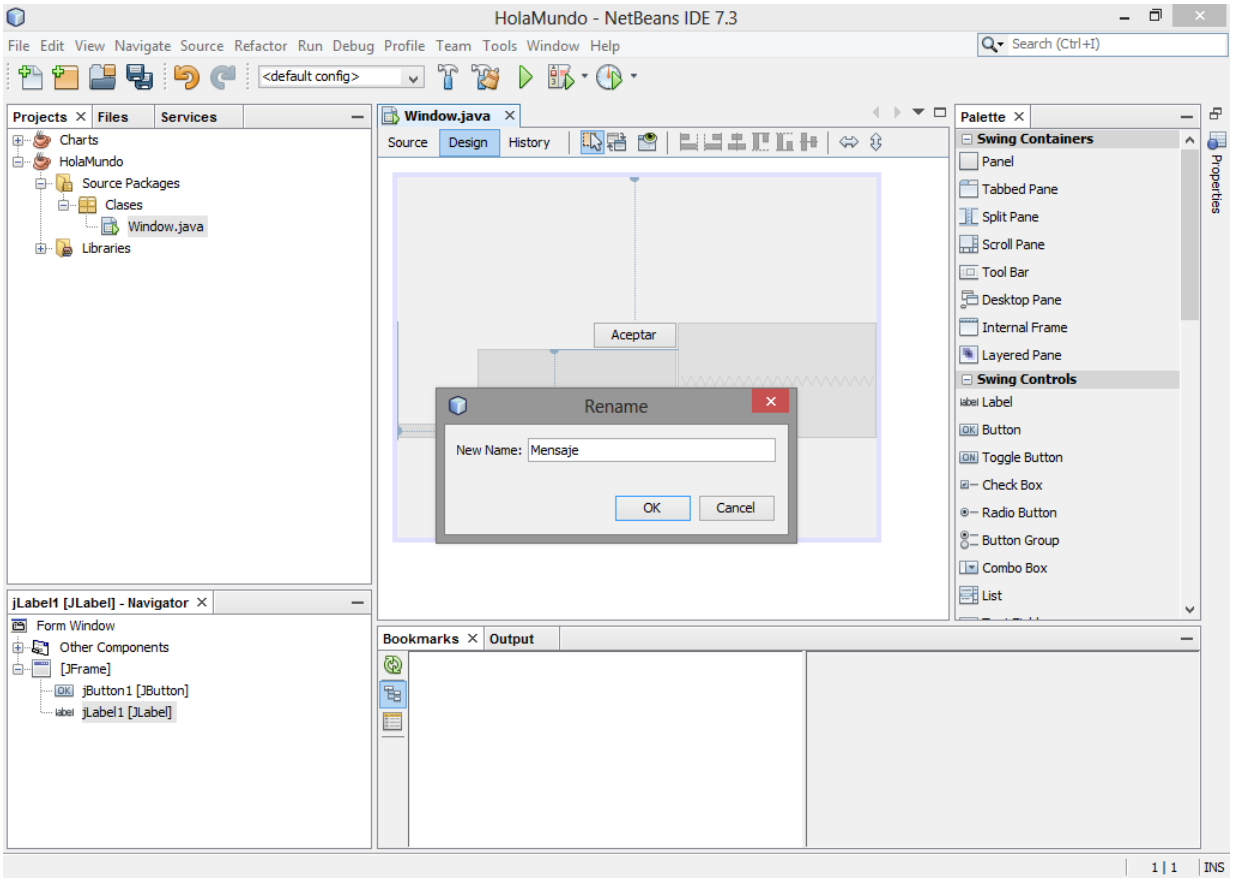


Ahora buscamos la opción llamada «Label» y la arrastramos al Frame.

Le damos clic derecho al botón que agregamos y le damos clic a «Editar Texto» o «Edit Text».

Escribimos «Aceptar».

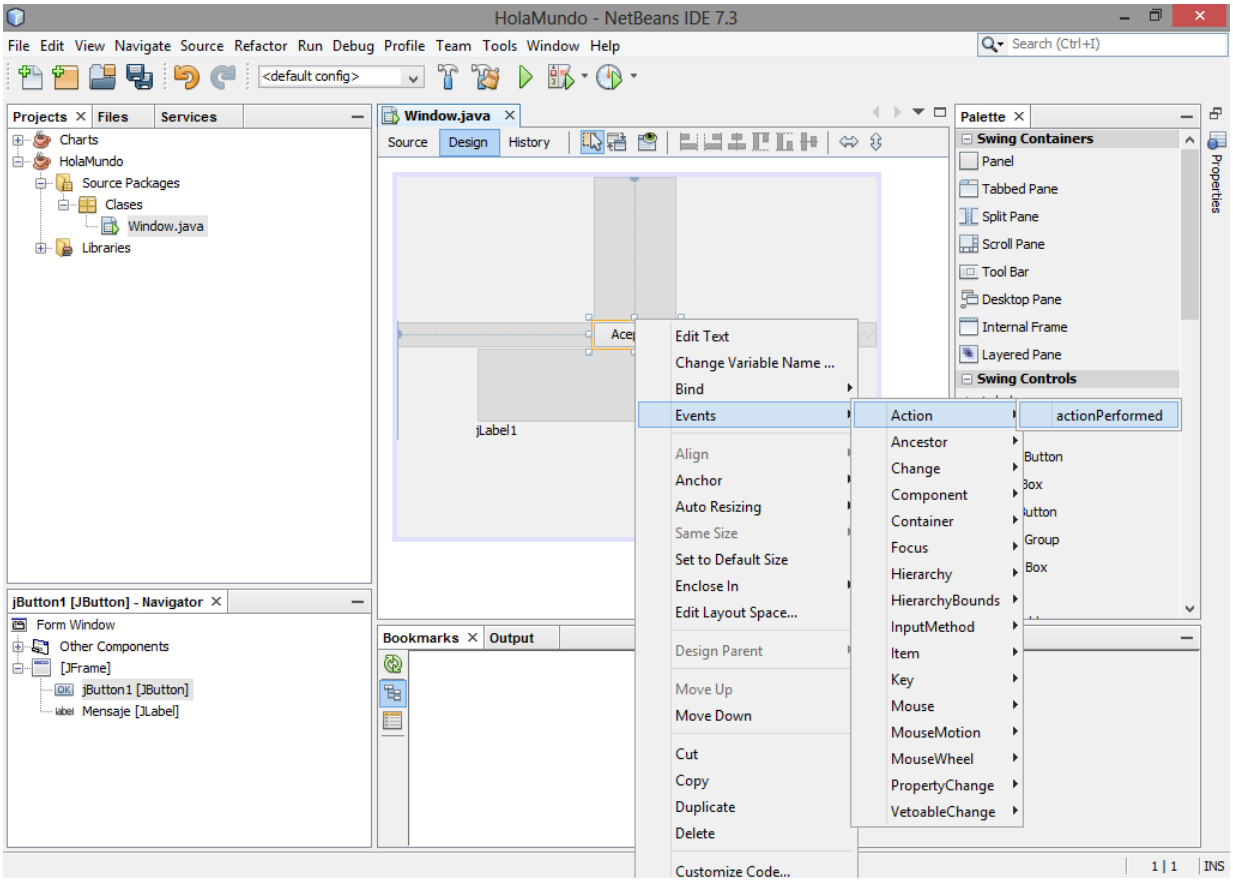
Ahora hacemos lo mismo con el Label, esta vez lo llamaremos «Mensaje».



Esta será la interfaz de nuestra primera aplicación. En la sección de la derecha, en Propiedades, es posible cambiar la fuente, el tipo de letra, el tamaño, etc.

Ahora procederemos a programar el comportamiento de nuestra aplicación.

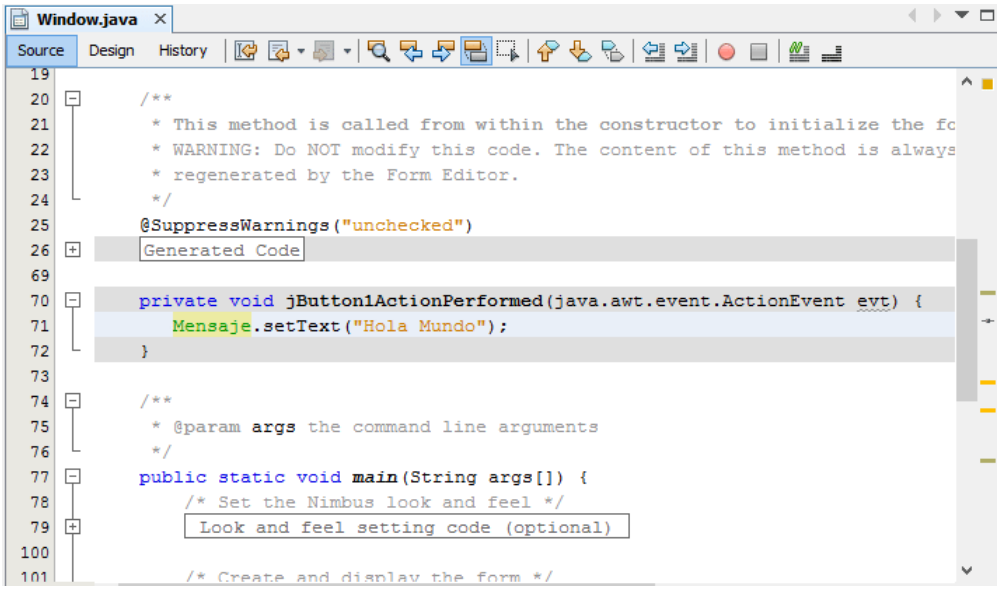
Para ello, hacemos clic derecho en el botón que colocamos y vamos a Eventos/Events, Acciones/Action, **actionPerformed**.



Se nos abrirá una ventana de códigos en la cual designaremos el comportamiento del botón que colocamos en el Frame.

Eliminamos el mensaje que nos aparece por default («// TODO add your handling code here:») y escribimos lo siguiente:

```
1 jLabel1.setText("Hola Mundo");
```



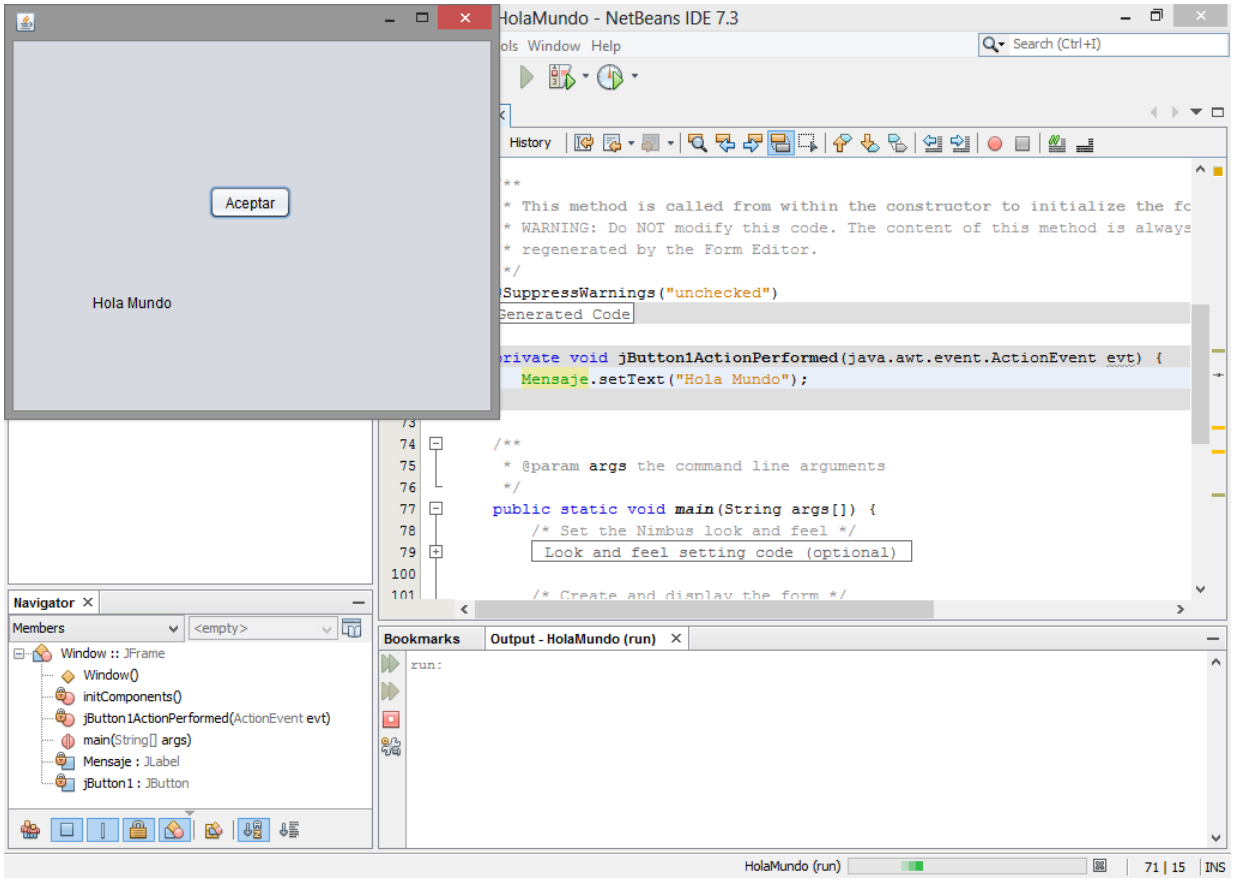
Explicaré lo que esto significa:

jLabel1 es el nombre del label que colocamos, el mismo al que le escribimos «Mensaje».

Con el comando **setText()** establecemos cualquier texto que deseemos. El texto que vayamos a designar debe colocarse entre comillas.

Luego que hemos escrito esto, podemos proceder a probar nuestro programa.

Ejecutamos el programa presionando F6 y nos aparecerá una ventana con un botón y la inscripción «Mensaje». Al hacer clic en el botón «Aceptar» vemos que donde decía «Mensaje» nos aparecerá «Hola Mundo».



6. Después de realizada las actividades se deben exportar cada carpeta completa del ejercicio, comprimirlas y publicarlas en la plataforma Edmodo, de forma individual.

Recursos:

- Netbean: https://netbeans.org/index_es.html
- Java software: https://www.youtube.com/watch?v=L1oMLsiMusQ
- Enlace tutorial: http://panamahitek.com/java/
- Curso1 en Java: https://www.codecademy.com/courses/learn-java/lessons/hello-world-java/exercises/introduction-to-java
- Enlace Tutorial: http://panamahitek.com/que-son-las-clases-en-java/
- Curso 2 en Java: https://codigofacilito.com/cursos/JAVA
- Sitio Web del Docente: https://rgloriacecilia.wixsite.com/mediatecvillaflora
- Plataforma Edmodo: https://new.edmodo.com/groups/grado-101-2020-31028086 (ingreso usuario y contraseña).

	INSTITUCIÓN EDUCATIVA VILLA FLORA	CÓDIGO: ED-F-30	VERSIÓN 2
	Taller 3	FECHA: 23-02-2019	

Marque el tipo de taller: Complementario _____ Permiso _____ Desescolarización _____ Otro x
Asignatura: Media Técnica Grado: 11°1 Fecha: Marzo 24/2020

Docente: Gloria Cecilia Ríos Muñoz
Nombre y Apellidos de estudiante:
_____

Propósito (indicador de desempeño):

- Procedimental: Utiliza herramientas informáticas de acuerdo con necesidades de manejo de información
- Actitudinal: Trabaja en Equipo respetando y valorando a sus compañeros

Pautas para la realización del taller:

1. Seguir los pasos dados
2. Elaborar al investigación en el cuaderno
3. Seleccionar su compañero de trabajo (Se evalúa trabajo en equipo, no puede ser individual).
4. Crear (Uno de los integrantes) en el sitio Google Drive, una carpeta llamada Media Técnica y dentro de ella la presentación que van a trabajar.
5. Elaboración de una presentación colaborativa de forma virtual (parejas), que cumpla con los siguientes parámetros, también se comparte video donde emiten otras pautas:
    - Apoyarse en fundamentos sólidos y sencillos.
    - Poco texto e imágenes contundentes
    - No fatigue a los asistentes con demasiada información
    - Utilice el contraste de colores.
    - Incluya imágenes y gráficos novedosos que permitan la ambientación
    - Tener en cuenta la Buena ortografía
    - Utilice la herramienta que desee (Ver algunas en los recursos suministrados)
    - Aplique Transiciones y Animaciones suaves.
6. Compartir la presentación a su compañero y a la docente en el siguiente correo electrónico: gloria.rios@ievillaflora.edu.co
7. Preparar la Exposición, no llegar a improvisar

Ítems de Evaluación:

- Elaboración de la presentación 100% Indicador Procedimental.
- Exposición 100% Indicador Actitudinal.

Actividades:

1. Conceptualización - **TEMA: Entornos de Desarrollo. Iniciamos con el proceso de** Investigación de los siguientes temas: (Registra en tu cuaderno).

    a) Definición de un entorno de desarrollo. Se comparte enlace en los recursos
    b) Explica los niveles de un entono de desarrollo

c) Característica de un entorno de desarrollo
d) Lista 5 entornos de desarrollo que conozcas en relación a Programación
e) Define que es un IDE y algunos ejemplos
f) Describe cuál es el entorno de desarrollo de Java
g) Describe cuál es el entorno de desarrollo web

2. Luego de la investigación y de extraer los elementos claves debes elaborar una Presentación utilizando una de las herramientas dadas. Recuerda que puedes unirte con otro compañero para que de forma colaborativa y virtual definan y elaboren la presentación.
   Antes de dar inicio observe el siguiente enlace con reglas para realizar una buena presentación: https://www.youtube.com/watch?v=PbMAU19tULM
   También un Video para presentaciones visuales: https://www.youtube.com/watch?v=bUHR7lYepT0

3. Selección de la herramienta a utilizar
4. Elaboración de la presentación
5. Luego de elaborar la presentación, compartir la presentación con el compañero y la docente en el siguiente correo electrónico: gloria.rios@ievillaflora.edu.co
6. Prepara la exposición para el próximo encuentro. Recuerde "Prepara No improvisar"

Recursos:

- Cuaderno físico
- Google Drive: Compartir la presentación en la carpeta de Media Técnica
- Reglas para realizar una buena presentación: https://www.youtube.com/watch?v=PbMAU19tULM
- Video para presentaciones visuales: https://www.youtube.com/watch?v=bUHR7lYepT0
- Internet
- Pc
- Enlace: https://www.arimetrics.com/glosario-digital/entorno-de-desarrollo

Herramientas que pueden utilizar para la presentación:

1.- Prezi.
2.- Knovio.
3.- Emaze.
4.- PowToon.
5.- Genial.ly.
6.- Sway.
7.- Canva – **Presentaciones**.
8.- Visme
9. Power Point.

	INSTITUCIÓN EDUCATIVA VILLA FLORA	CÓDIGO: ED-F-30	VERSIÓN 2
	Taller 4	FECHA: 23-02-2019	

Marque el tipo de taller: Complementario _____ Permiso _____ Desescolarización _____ Otro x
Asignatura: Tecnología e Informática Grado: 11°1 Fecha: Marzo 25 /2020

Docente: Gloria Cecilia Ríos Muñoz
Nombre y Apellidos de estudiante:
_____

Propósito (indicador de desempeño):

- Procedimental: Crea las tablas y objetos de la base de datos de acuerdo con el diseño y el motor de Base de datos.
- Actitudinal: Interpreta la información técnica de diseño para la codificación del software

Pautas para la realización del taller:

1. Seguir los pasos dados
2. Elaborar las investigaciones de toda la actividad en el cuaderno y el test
3. Las actividades corresponde a procesos y avances de tu proyecto de Media Técnica, después se selecciona el mejor. Todos deben hacer el proceso.
4. Elaborar el Informe completo en Word archivo digital de acuerdo a los pasos solicitados, tareas y aportes
5. Elaborar la Base de datos de tu proyecto y el formulario en Xampp
6. Trabajar en parejas
7. Publicar actividades en Edmodo

Ítems de Evaluación:

- Cuaderno Físico investigaciones y Solución del Test 40% Indicador Procedimental.
- Elaboración de la Base de Datos y el formulario 60% Indicador Procedimental
- Informe escrito en Word. 100% Indicador Actitudinal.

Actividades

1. Retomando los conceptos previos y vistos en clase deben afianzarlos y definir los elementos (Tablas, Formularios), que van hacer parte de la Base de Datos de sus proyectos.

2. Observar en el siguiente ejemplo, se comparte las tablas correspondientes a la Base de datos de un Hospital y sus relaciones, algo similar deben presentar en esta actividad. También la de una Biblioteca.

   Tener muy claro cómo va a desarrollar su propuesta. (Pueden Ampliar las imágenes para visualizarlas mucho mejor).

Ejemplo Tablas Base de Datos Hospital.

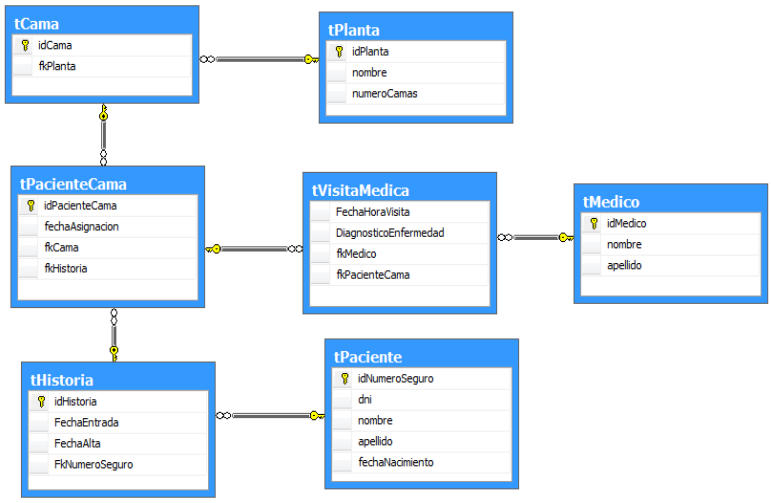
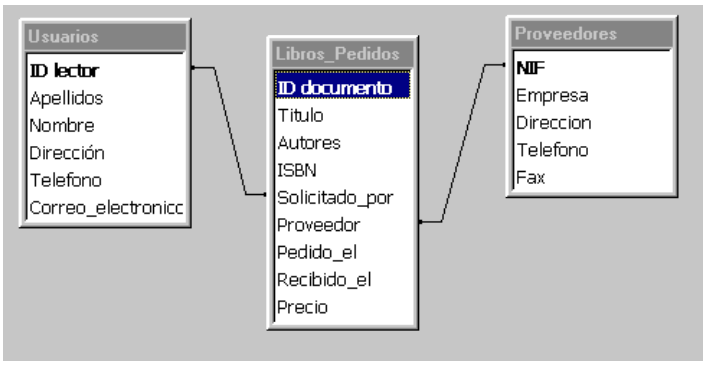


Tabla Base de Datos Biblioteca



3. Consultar ¿Qué es una clave principal? , ¿Qué es una clave foránea?,¿ Qué son las relaciones en las bases de datos? y ¿Qué es normalización? Y ¿Qué es cardinalidad?. Para ser aplicadas en la Base de Datos de su proyecto.

4. Se agrupan en parejas.

5. Inician con el diseño de las tablas de la Base de datos de sus proyectos
6. Luego en un documento en Word preparan el informe teniendo en cuenta los siguientes pasos:

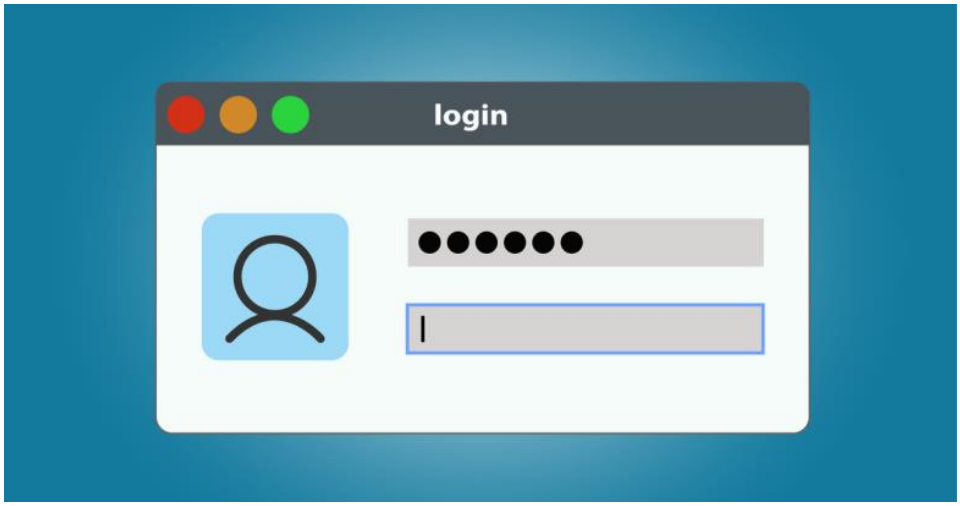
   a) Portada con Datos integrantes
   b) Nombre del proyecto
   c) Descripción del problema
   d) Objetivo general
   e) Las tablas a utilizar con su nombre, campos, datos y tipos de datos.
   f) ¿Cuáles tablas se van a relacionar?
   g) Imagen de la Base de Datos completa
   h) ¿Qué tablas consideras se debe relacionar?

1. Luego en Xampp elaboran la Base de datos, capturan los pantallazos de la BD y los agregan al informe. Comparto en los Recursos documento del paso a paso y material para que puedes apoyarte en la elaboración de la Base de Datos.

8 Luego diseñan el Formulario que deseen utilizar para sus bases de datos. Comparto ejemplo del formulario a utilizar en la Biblioteca.

Título	La Regenta /
Autor	Alas, Leopoldo
Editorial	Madrid : Alianza Editorial, D.L. 2014.
Edición	
Fecha de publicación	2014
ISBN/ISSN	978-84-206-9160-2
Otra información	

9 Por último Consultan la programación de un formulario inicio de sesión.



10 Registra al final del informe los aportes (Mínimo 3 por persona) más relevantes que lograste de cada tema enunciado.

11 Responder el siguiente Test en el cuaderno, que permite validar tus conocimientos adquiridos.

Para cada una de las siguientes cuestiones elige razonadamente cada una de las respuestas correctas.
¿Cuáles de los siguientes puntos representan inconvenientes de los Sistemas de Base de Datos?
a. Redundancia e Inconsistencia.
b. Sistema de Gestión de Datos independiente de la máquina y del SO.
c. Control de concurrencia.
d. Difícil modificación de los datos.
Los sistemas orientados a BD presentan las siguientes ventajas…
a. Integridad de los datos.
b. Redundancia.
c. Cada aplicación maneja sus propios datos.
d. Independencia entre los datos y las aplicaciones que los usan.
Los datos son …
a. … todo aquello de lo cual interesa guardar información.
b. … hechos conocidos que pueden registrarse y que tienen un significado implícito.
c. … información acerca de los metadados.
d. … las claves primarias y foráneas de cada entidad.
Un SGBD …
a. … esta formado por datos acerca de los datos presentes en la base de datos.

b. … es una aplicación que permite a los usuarios definir, crear y mantener una base de datos, y proporciona acceso controlado a la misma.
c. … permite a los usuarios tener acceso a la BD completa impidiendo restricciones.
d. … permite la inserción, actualización, eliminación y consulta de datos mediante el lenguaje de manejo o manipulación de datos.
Definir que es un SGBD
Indica las principales ventajas de las BD frente a los antiguos sistemas de ficheros.
Nombre los distintos tipos de bases de datos que existen según el modelo que siguen
¿Qué son las vistas? ¿Para qué se utilizan?. Busca información en Internet para completar tu respuesta. ¿En qué se diferencia de una consulta?
Describe el significado de las siguientes siglas: DDL, DML y DCL. Explica la utilidad de cada una.
¿Qué es un script o guión?
Recuerda y define los siguientes conceptos:
1. Dato
2. Tipo de Dato
3. Campo
4. Registro
5. Tabla
6. Consulta
7. Procedimiento
¿Qué es el diccionario de datos?
¿Qué quiere decir que una base de datos permita la concurrencia?

12 Socialización en la próxima clase de toda la actividad.

Recursos:

- Cuaderno Físico
- Internet
- PC
- Plataforma Edmodo
- Xampp
- Enlace Tutorial: https://gestionbasesdatos.readthedocs.io/es/latest/Tema2/Teoria.html
- Enlace Documento B d D: https://pensamientosaztlek.files.wordpress.com/2013/07/phpmyadmin_crearunabasededatosmysqldesdephpmyadmin.pdf
- Video 1: http://www.mclibre.org/consultar/php/otros/xampp-instalacion -windows.html
- Video 2: https://www.youtube.com/watch?v=9zAVE-ImLbY
- Video 3: https://www.youtube.com/watch?v=Qz_VBz6ldlk Instalador de Xampp: