

NOTA: Antes de empezar con este tutorial quiero decir que todo lo que sé es gracias a un canal de YouTube llamado “pildorasinformaticas”, os dejo un link a su lista de reproducción sobre Java: <https://www.youtube.com/playlist?list=PLU8oAIHdN5BktAXdEVCLUYzvDyqRQJ2Ik>

He de decir que yo voy a utilizar el entorno de desarrollo Eclipse, pero tal vez algún que otro profesor os obligue a utilizar DrJava, para ver como instalarlo correctamente mirad el PDF0.

La sintaxis es la misma en ambos entornos, de momento que sepa solo se diferencia en la forma de crear paquetes de clases, cosa que veremos en el PDF2.

Comencemos...

Una variable es una referencia a un espacio de memoria que contiene una información de un tipo, hay dos clases de tipos, los *primitivos* y los *no primitivos* (objetos), explicaré esto de los *no primitivos* al llegar al PDF de “Clases y Objetos”, los tipos que usaremos son:

PRIMITIVOS:

-boolean -> Puede tener dos valores *true* o *false*.

-int -> Números enteros (-1512, -432, -12, -1, 0, 1, 21, 64262)

-double -> Números con decimales (-31.6236, -12, -0.532, 0, 0.532, 3.1415, 14)

char -> Contienen un carácter en código ASCII ('a', 'x', '*', '5')

NO PRIMITIVOS:

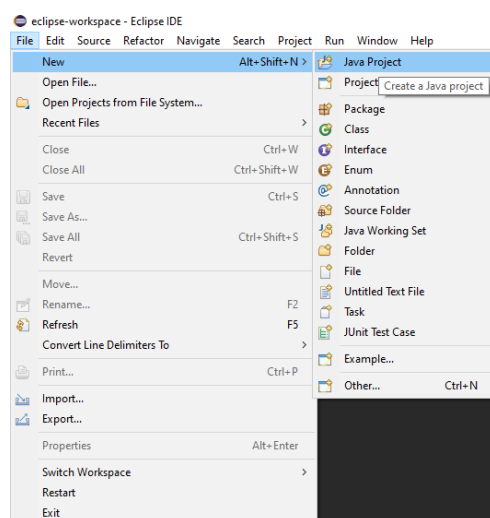
-String -> Cadenas de caracteres (“Hola Mundo.”, “Esto es una prueba”);

-Double -> Números con decimales (-3.1415, -0.42, 0.0, 3.0, 14.312); Estos tienen que llevar un punto obligatoriamente

Si vais a utilizar DrJava, os recomiendo crear una carpeta donde tengáis todo **ORDENADO**, que luego os aseguro que es un caos encontrar vuestros labs y proys.

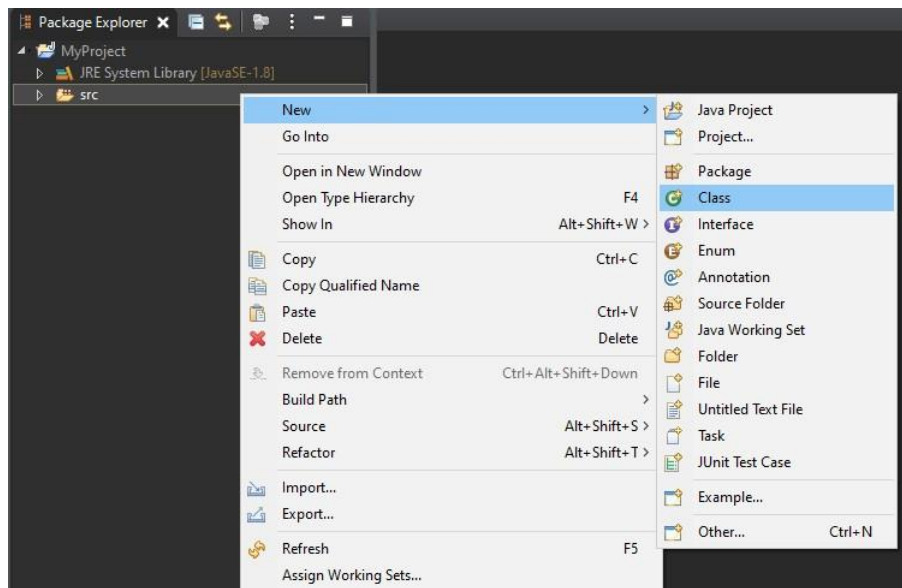
COMO CREAR UNA CLASE

Eclipse

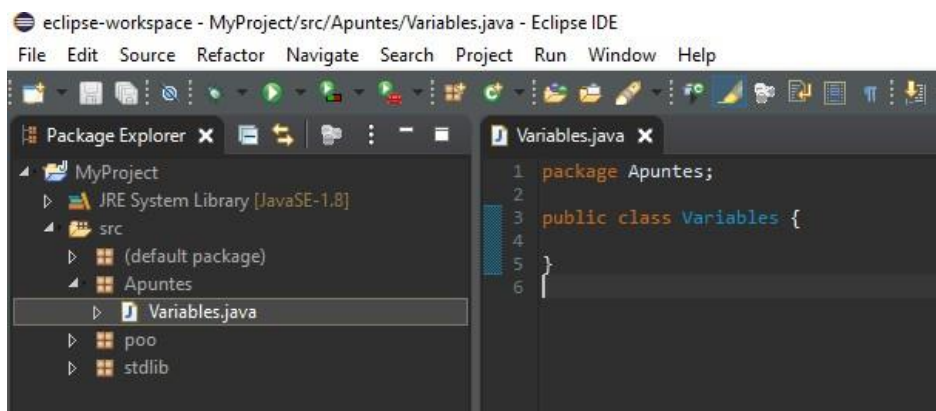


Ahora en “Project name” ponemos el nombre que queramos, y finish.

Ahora encima de la carpeta src, creamos una nueva clase, el nombre de nuestra clase **tiene** que ir todo junto y tener la primera letra mayúscula.



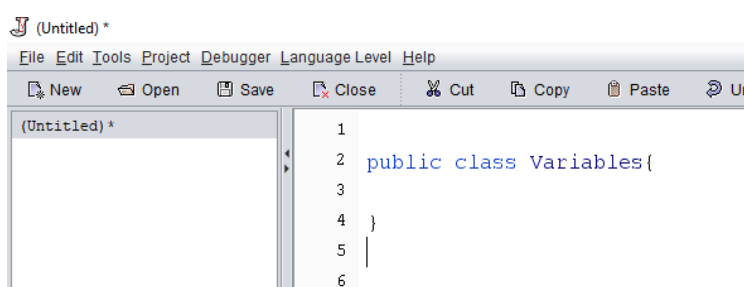
Eclipse nos pone automáticamente la estructura de la clase. (yo la he llamado variables)



Vosotros no tendréis lo de *package* en la primera línea, eso se debe a que yo he creado un nuevo paquete para guardar todo lo que haré en esta guía, y todas las clases que no estén en el paquete (*default package*) tienen que llevar esa sentencia para indicar a que paquete pertenecen, ya os lo explicaré en el PDF2.

DrJava

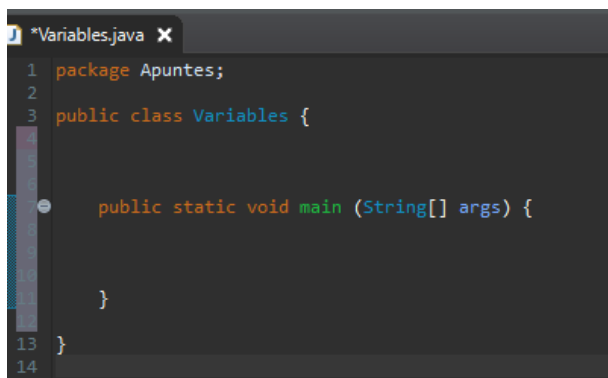
Solo escribiremos estas líneas en archivo nuevo.



Para ir creando clases, le daremos al botón *New*, que se ve en la imagen anterior, arriba a la izquierda. Para guardar estas clases tendremos que indicarles la carpeta donde se guardarán, de ahí que sea muy importante la organización, para guardar la clase solo haz click en *Compile* en la barra de arriba e indica la carpeta donde debe guardarse.

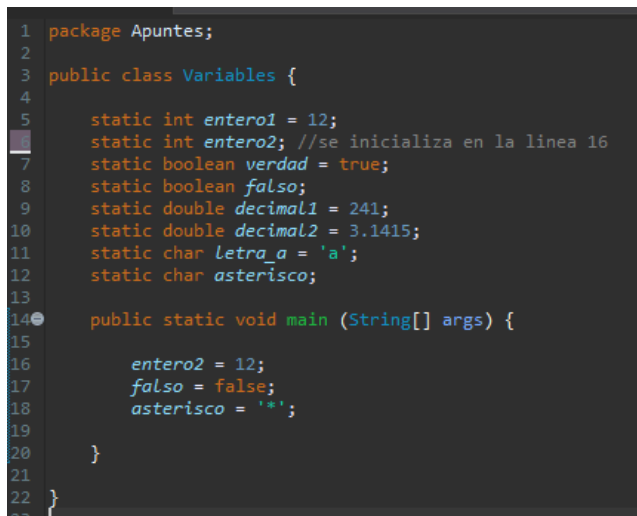
A partir de aquí continuaremos con **solamente Eclipse**, pero todo es aplicable a DrJava directamente.

Dentro de nuestra clase crearemos la función *main*, al ejecutar el programa solo se realizará lo que haya dentro de esta función:



```
1 package Apuntes;
2
3 public class Variables {
4
5     public static void main (String[] args) {
6
7     }
8 }
9
```

Ahora bien, para crear variables lo haremos de la siguiente forma, voy a poner varias maneras:



```
1 package Apuntes;
2
3 public class Variables {
4
5     static int entero1 = 12;
6     static int entero2; //se inicializa en la linea 16
7     static boolean verdad = true;
8     static boolean falso;
9     static double decimal1 = 241;
10    static double decimal2 = 3.1415;
11    static char letra_a = 'a';
12    static char asterisco;
13
14    public static void main (String[] args) {
15
16        entero2 = 12;
17        falso = false;
18        asterisco = '*';
19
20    }
21 }
22
```

Si las creamos fuera de una función (como la *main*) tendremos que poner *static* delante para saber que no es un *atributo*, esto de “atributo” se explicara en el PDF de “Clases y Objetos”, de momento solo nos hace falta saber eso, crear una variable se realiza entre las líneas 5-12, pero **inicializarlas** se hace en las líneas 5,7,9,10,11,16,17,18. Es decir, crearlas es declarar que estás variables existen, definiendo su tipo de dato y nombre, mientras que inicializarlas es asignarles un valor, si no las inicializamos tendrán un valor por defecto, el cual es:

int = 0 , boolean = false , double = 0 , char = “ ” , String = null , Double = null

Como veis los tipos que empiezan por mayúscula (los *no primitivos* u *objetos*) se les asigna el valor *null*, esto se explicará en el PDF “Clases y Objetos”.

```

package Apuntes;

public class Variables {

    public static void main (String[] args) {

        int entero1 = 12;
        int entero2;
        entero2 = 12;
        boolean verdad = true;
        boolean falso;
        falso = false;
        double decimal1 = 241;
        double decimal2 = 3.1415;
        char letra_a = 'a';
        char asterisco;
        asterisco = '*';

    }

}

```

Como veis también se pueden crear dentro de una función, si se hace así, no llevarán el *static* delante, ahora veamos los tipos *no primitivos*.

```

1 package Apuntes;
2
3 public class Variables {
4
5     static String str1 = "Hola Mundo";
6     static Double doble1;
7     static Double doble2 = 0.0;
8
9     public static void main (String[] args) {
10
11         int entero = 12;
12         boolean verdad = true;
13         double decimal = 241;
14         char letra_a = 'a';
15
16         String str2 = "Esto es una prueba.";
17         doble1 = 3.14;
18
19     }
20
21 }

```

Se crean de la misma forma, recordad que los *Double* se diferencia de los *double* en que siempre llevan un punto, aunque quieras representar el 3.0, no puedes escribir simplemente 3.

OPERACIONES

Tenemos las siguientes operaciones.

-Suma -> +

-Resta -> -

-Multiplicación -> *

-División -> /

-Módulo (Resto de una división) -> %

Además de éstas utilizaremos una función de la clase *Math*, esto se explicará más adelante.

-Eleva a una potencia -> *Math.pow*(BASE, EXPONENTE)

Estas operaciones no solo se pueden hacer con números, también se pueden hacer con variables, pero hemos de tener cuidado con los tipos, ya que algunos no se pueden operar entre sí, en la siguiente tabla no importa el orden:

DATO 1	DATO 2	RESULTADO
boolean		No se pueden operar con ningún dato <i>primitivo</i>
int	Int	int
int	double	double
int	Double	Double
int	char	int
double	Double	double
double	char	double
char	Double	double
String	X	Solo se puede sumar → String

Algunas curiosidades son que cualquier cosa que le sumes a un String, este lo convertirá a String, y que un char es un *int*, el número será en que le corresponda en el código ASCII, por ejemplo, la letra 'a' es el 97, por lo que al sumarle 3 quedará 100.

Voy a poner unos ejemplos de operaciones.

```
"Variables.java" X
1 package Apuntes;
2
3 public class Variables {
4
5     static String str1 = "Hola Mundo";
6     static Double doble1;
7     static Double doble2 = 0.0;
8
9     public static void main (String[] args) {
10
11         int entero = 12;
12         boolean verdad = true;
13         double decimal = 241;
14         char letra_a = 'a';
15
16         String str2 = "Esto es una prueba.";
17         doble1 = 3.14;
18
19         str1 = str1 + entero; //"Hola Mundo12"
20         double num = doble1 - letra_a; //-93.86
21         int entero2 = entero / 5; //2
22
23     }
24
25 }
```

Los nombres de las variables tienen que tener la **primera letra minúscula** e ir todo junto, para separar palabras recomiendo poner la primera letra en mayúscula *nombreDeMiVariable*.

Ahora un par de asignaciones que son equivalentes:

```
int c = 0;
c++; // c = 1
c += 1; // c = 2
c = c + 1; // c = 3

int i = 100;
i *= 3; //i = 300
i = i * 10; //i = 3000;
i /= 3000; i = 1;
```

Como veis en el primer bloque, esas 3 líneas son **equivalentes**, es decir, `c++` es lo mismo que sumar 1 a una variable, se puede restar 1 poniendo `c--`. Además, podemos realizar otras asignaciones como se ve la antepenúltima y penúltima línea, estas son:

`+=` `-=` `*=` `/=` `%=`

Os recomiendo que probéis por vuestra cuenta como realizar estas operaciones y asignaciones, ya que tienen algunas curiosidades, por ejemplo, si dividimos un *int* entre un *int* nos dará un *int*, por lo que al dividir 12/5 nos queda 2, en vez de 2.4, ya que se queda solo con la parte entera, recomiendo estudiar un poco la tabla de arriba sobre operaciones.

Otra nota importante es que el resultado de la función *Math.pow()* siempre devuelve un tipo *double*, y que si queremos hayar la raíz cuadrada de un número podemos poner, *Math.pow(nuestroNumero,0.5)*, es decir, elevar nuestro número a 0.5, para la raíz cubica lo elevamos a 1 / 3.0, ya que como he dicho antes, si solo ponemos 1/3 nos dará 0, es decir, la parte entera.