Curso de Programación en JAVA

Ing. Alejandro Leyva

1. Introducción al Lenguaje

¿Qué es JAVA?

Java es un lenguaje de programación de propósito general, concurrente, orientado a objetos, compilado, multi hilo.

Permiten que los desarrolladores de aplicaciones escriban el programa una vez y lo ejecuten en cualquier dispositivo (conocido en inglés como WORA, o "write once, run anywhere").

¿Qué es JAVA?

Historia de JAVA

1995

JAVA

Sun Microsystems Inc. lanza JAVA 1.0

2006

JAVA SE 6

Se lanza la versión JAVA 6 (Mustang). La cual se mantendría hasta 2011 con la versión 7

2009

Compra de Sun Microsystem por Oracle

Oracle hace la adquisición de Sun Microsystem el 20 de abril de 2009.

2018

JAVA 10

Al día de hoy tenemos la versión JAVA 10.

¿Por qué JAVA?

Multiplataforma

El código compilado (bytecode) es ejecutado en una máquina virtual (JVM)

Compilado

JAVA es un lenguaje compilado, esto lo hace rápido y seguro para cualquier tipo de aplicación.

Android

Si deseas entrar al desarrollo móvil, debes saber JAVA.

¿Por qué JAVA?

Jun 2018	Jun 2017	Change	Programming Language	Ratings	Change
1	1		Java	15.368%	+0.88%
2	2		C	14.936%	+8.09%
3	3		C++	8.337%	+2.61%
4	4		Python	5.761%	+1.43%
5	5		C#	4.314%	+0.78%
6	6		Visual Basic .NET	3.762%	+0.65%
7	8	^	PHP	2.881%	+0.11%
8	7	v	JavaScript	2.495%	-0.53%
9	-	^^	SQL	2.339%	+2.34%
10	14	^^	R	1.452%	-0.70%

https://www.tiobe.com/tiobe-index/

¿Por qué JAVA?

https://insights.stackoverflow.com/survey/2018/

¿Por qué JAVA?

Programming languages

https://www.jetbrains.com/research/devecosystem-2017/

Proyectos que están desarrollados en JAVA

NetBeans

PostgreSQL

SAT

2. Configuración del Entorno de Desarrollo

Entorno de desarrollo Integrado (IDE)

---

http://www.oracle.com/technetwork/java/javase/downloads/index.html

NetBeans IDE 8.2 Download

8.1 | 8.2 | Development | Archive

Email address (optional):

Subscribe to newsletters: ☑ Monthly ☐ Weekly

☑ NetBeans can contact me at this address

IDE Language: English

Platform: Linux (x86/x64)

Note: Greyed out technologies are not supported for this platform.

NetBeans IDE Download Bundles

Supported technologies *	Java SE	Java EE	HTML5/JavaScript	PHP	C/C++	All
NetBeans Platform SDK	●	●				●
Java SE	●	●				●
Java FX	●	●				●
Java EE		●				●
Java ME						●
HTML5/JavaScript		●	●	●		●
PHP			●	●		●
C/C++					●	●
Groovy						●
Java Card™ 3 Connected						–
Bundled servers						
GlassFish Server Open Source Edition 4.1.1		●				●
Apache Tomcat 8.0.27		●				●
	Download	Download	Download x86 / Download x64	Download x86 / Download x64	Download x86 / Download x64	Download
	Free, 94 MB	Free, 196 MB	Free, 116 - 119 MB	Free, 116 - 119 MB	Free, 115 - 117 MB	Free, 214 MB

* You can add or remove packs later using the IDE's Plugin Manager (Tools | Plugins).

HTML/JS, PHP and C/C++ NetBeans bundles include Java Runtime Environment and do not require a separate Java installation.

JDK 8 is required for installing and running the Java SE, Java EE and All NetBeans Bundles. NetBeans 8.2 does not run on JDK9! You can download a standalone JDK or download the latest JDK with NetBeans IDE Java SE bundle.

Important Legal Information:

NetBeans Community Distributions are available under a Dual License consisting of the Common Development and Distribution License (CDDL) v1.0 and GNU General Public License (GPL) v2. Such distributions include additional components under separate licenses identified in the License file. See the Third Party License file for external

Java SE
Java EE
Java ME
Java SE Advanced & Suite
Java Embedded
Java DB
Web Tier
Java Card
Java TV
New to Java
Community
Java Magazine

Overview | Downloads | Documentation | Community | Technologies | Training

JDK 8u171 with NetBeans 8.2

This distribution of the JDK includes the Java SE bundle of NetBeans IDE, which is a powerful integrated development environment for developing applications on the Java platform. Learn more

You must accept the JDK 8u171 and NetBeans 8.2 Cobundle License Agreement to download this software.

Accept License Agreement Decline License Agreement

Java SE and NetBeans Cobundle (JDK 8u171 and NB 8.2)		
Product / File Description	**File Size**	**Download**
Linux x86	296.49 MB	jdk-8u171-nb-8_2-linux-i586.sh
Linux x64	291.4 MB	jdk-8u171-nb-8_2-linux-x64.sh
Mac OS X x64	363.44 MB	jdk-8u171-nb-8_2-macosx-x64.dmg
Windows x86	327.08 MB	jdk-8u171-nb-8_2-windows-i586.exe
Windows x64	338.65 MB	jdk-8u171-nb-8_2-windows-x64.exe

- License
- Java SE 8 Readme
- NB 8.2 3rd Party Readme
- Installation Instructions
- Java SE Release Notes
- NetBeans Release Notes

Java SDKs and Tools
Java SE
Java EE and Glassfish
Java ME
Java Card
NetBeans IDE
Java Mission Control

Java Resources
Java APIs
Technical Articles
Demos and Videos
Forums
Java Magazine
Developer Training
Tutorials
Java.com

Java SE
Java EE
Java ME
Java SE Advanced & Suite
Java Embedded
Java DB
Web Tier
Java Card
Java TV
New to Java
Community
Java Magazine

Overview | Downloads | Documentation | Community | Technologies | Training

JDK 8u171 with NetBeans 8.2

This distribution of the JDK includes the Java SE bundle of NetBeans IDE, which is a powerful integrated development environment for developing applications on the Java platform. Learn more

You must accept the JDK 8u171 and NetBeans 8.2 Cobundle License Agreement to download this software.

Thank you for accepting the JDK 8u171 and NetBeans 8.2 Cobundle License Agreement; you may now download this software.

Java SE and NetBeans Cobundle (JDK 8u171 and NB 8.2)		
Product / File Description	**File Size**	**Download**
Linux x86	296.49 MB	jdk-8u171-nb-8_2-linux-i586.sh
Linux x64	291.4 MB	jdk-8u171-nb-8_2-linux-x64.sh
Mac OS X x64	363.44 MB	jdk-8u171-nb-8_2-macosx-x64.dmg
Windows x86	327.08 MB	jdk-8u171-nb-8_2-windows-i586.exe
Windows x64	338.65 MB	jdk-8u171-nb-8_2-windows-x64.exe

- License
- Java SE 8 Readme
- NB 8.2 3rd Party Readme
- Installation Instructions
- Java SE Release Notes

Java SDKs and Tools

- Java SE
- Java EE and Glassfish
- Java ME
- Java Card
- NetBeans IDE
- Java Mission Control

Java Resources

- Java APIs
- Technical Articles
- Demos and Videos
- Forums
- Java Magazine
- Developer Training
- Tutorials
- Java.com

Configuración del PATH (Windows)

Configuración del PATH (Windows)

Configuración del PATH (Windows)

Configuración del PATH (Windows)

Configuración del PATH (Windows)

JAVA_HOME

Configuración del PATH (Windows)

;%JAVA_HOME%\bin

Proceso de compilación y ejecución

JAVA VIRTUAL MACHINE

Para poder ejecutar nuestro programa, se toma el archivo ByteCode, el cual es pasado a la JVM, la cual interpreta el archivo compilado.

Archivo .java

Creamos nuestro archivo fuente en un editor, como Netbeans

Compilación

Nuestro archivo fuente .java pasa por el compilador, se verifica y si es correcto se compila. Generando un archivo **ByteCode**

Nuestro primer “**Hola mundo**”

```java
public class HolaMundo {

    public static void main(String[] args) {
        System.out.println("Mi primer hola mundo");
    }

}
```

HolaMundo.java

Programa en JAVA

Nombre de la clase

```
public class HolaMundo {

    public static void main(String[] args) {
        System.out.println("Mi primer hola mundo");
    }
}
```

Método main

Sentencia ejecutada cuando el programa está corriendo

Toda sentencia termina con punto y coma (;)

Buenas prácticas

Camel Case

Identificadores

- **Sólo puede contener**
  - Letras (A,B,C...Z)
  - Números (0,1,2,...9)
  - Guion bajo (_)
  - Signo de peso ($)

➢ **No puede**
  - Comenzar con número
  - Tener espacios
  - Tener acentos

- *Si es una clase, debe comenzar con Mayúscula, de lo contrario será con minúscula, ejemplo: HolaMundo.java*
- *Sensible a mayúsculas y minúsculas*

Secuencias de Escape

Nombre	Sintaxis
Siguiente línea (Enter)	\n
Tabulador	\t
Retorno de carro	\r
Diagonal invertida	\\
Doble comilla	\”

SecuenciasEscape.java

¿Qué es una variable?

Es un espacio en memoria, que se usa para almacenar un valor.

Tipos de variables (Datos primitivos)

Nombre	Sintaxis	Rango de valores
booleano	boolean	True, false
byte	byte	-128 a 127
Entero corto	short	-32,768 a 32,767
Entero	int	-2,147,483,648 a 2,147,483,649
Entero largo	long	$-9x10^{18}$ a $9x10^{18}$
Doble	double	$-1.79x10^{308}$ a $1.79x10^{308}$
Flotante	float	$-3.4x10^{38}$ a $3.4x10^{38}$
Caracter	char	caracteres

TiposVariables.java

Declaración de variables

```
tipoDeVariable nombreVariable; //declaración de variable

tipoDeVariable nombreVariable = valorAsignado;
```

```
int miVariableEntera = 5;

double miVariableDoble = 4.32;
```

Operadores Aritméticos

Nombre	Símbolo	Descripción
Asignación	=	Asignar un valor dado
Suma	+	Operación suma
Resta	-	Operación resta
Multiplicación	*	Operación producto
División	/	Operación división
Residuo	%	Retorna el residuo de la división

OperadoresAritmeticos.java

Operadores Aritméticos combinados

Nombre	Símbolo	Descripción
Suma	+=	x = x + 3 -> x+=3
Resta	-=	x = x - 3 -> x-=3
Multiplicación	*=	x = x * 3 -> x*=3
División	/=	x = x / 3 -> x/=3
Residuo	%=	x = x % 3 -> x%=3
Incremento	++	x = x + 1 -> x++
Decremento	--	x = x - 1 -> x--

OperadoresCombinados.java

Ejercicio de Física - Segunda Ley de Newton

Calcular la fuerza si un objeto tiene una aceleración de $10m/s^2$ y una masa de 2.5kg. Después, si el objeto incrementa su aceleración a $12.6m/s^2$.

$$F = m\ a$$

SegundaLey.java

Ejercicio - Conversiones

- Realizar programa para conversión de unidades, de centímetros a pulgadas.

2.54 cm = 1 inch

- Hacer la conversión de presión de bar -> N/mm^2.

1 bar = 0.1 N/mm^2

Conversiones.java

Método **print**

- ***print***(*“Texto”*); //impresión básica
- ***println***(*“Texto”*); //impresión con salto de línea
- ***printf***(*“Texto”*); //impresión con formato

Ejemplo de printf(); es decir, impresión con formato

```
double peso = 85.3656;

printf(“Mi peso es %.2f kgrs”, peso);//impresión con formato
```

-> Mi peso es 85.37 kgrs <- Salida

Printf - Especificadores de formato

Caracter	Tipo de salida	Ejemplo
d	Entero	%d %5d
f	Flotantes y dobles	%f %2.2f
e	Con notación científica	%8.3e e
s	String (Texto)	%s %12s
c	Caracter	%c %2c

3. Estructuras de Control y Decisión

Sentencia de decisión IF

```
if(condicionVerdadera){
    //en caso que sea verdadero, ejecuta éste código
}
if(5 >= 4){
    System.out.println(“5 es mayor o igual a 4”);
}
```

DesicionIf.java

Operadores de relación

Operador	Descripción	Ejemplo	Resultado
==	Igual que	8 == 9	false
<	Menor que	9 < 4	false
>	Mayor que	0 > -4	true
<=	Menor o igual que	9 <= 20	true
>=	Mayor o igual que	3 >= 6	false
!=	Diferente de	4 != 4	false

OperadoresRelacion.java

Ejercicio - Aprobado-Reprobado

Hacer un programa que nos indique si el alumno aprobó o reprobó la materia.

AprobadoReprobado.java

Ejercicio - Conociendo si es número es par o impar

Realizar un programa que diga si el número es par o impar y si el número es mayor 10, que diga un mensaje que el dígito dado es superior a 10.

ParImpar.java

Operadores lógicos

AND	
Operación	**Resultado**
False **&&** False	False
False **&&** True	False
True **&&** False	False
True **&&** True	True

OR	
Operación	**Resultado**
False **||** False	False
False **||** True	True
True **||** False	True
True **||** True	True

NOT	
Operación	**Resultado**
!True	False
!False	True

Operadores lógicos

Nombre	Símbolo	Aplicación	Resultado
AND	&&	(5 == 5) **&&** (4==4)	True
OR	||	(9 > 3) **||** false	True
NOT	!	!false	True

OperdadoresLogicos.java

Ejercicio - Aprobado-Reprobado con mensaje

Realizar un programa que diga una frase dependiendo de su calificación.

- Si obtuvo menos de 6 -> *“Nos vemos en recursa”*
- Si obtuvo de 6 hasta menos de 7-> *“De panzazo”*
- Si obtuvo de 7 hasta menos de 8 -> *“Echale más punch”*
- Si obtuvo de 8 hasta menos de 9 -> *“Bien, puedes mejorar”*
- Si obtuvo de 9 hasta menos de 10 -> *“Muy bien, te falto tantito”*
- Si obtuvo 10 -> *“Excelente, con toda la actitud”*
- *Si da otro valor que no esté definido dirá “No es posible”*

CalificacionesMensaje.java

Leyendo datos del teclado

Se importa el objeto Scanner, se genera una instancia.

```
import java.util.Scanner; //se importa la librería

Scanner leer = new Scanner(System.in); //crea instancia

int entero = leer.nextInt(); //lee y guarda entero

double doble = leer.nextDouble();//lee y guarda doble
```

AprobadoReprobado2.java, ParImpar2.java, CalificacionesMensaje2.java

Ejercicio - Ley de Ohm

Crear un menú dando las opciones para seleccionar que se desea calcular. Opciones: 1. Voltaje, 2. Corriente, 3 Resistencia y al final arrojar el resultado de la operación.

LeyOhmIf.java

Ejercicio - Neumática, Cilindro doble efecto

Calcule la fuerza efectiva que ejerce un cilindro de doble efecto en las carreras de *avance* y *retroceso* sabiendo que el émbolo tiene un diámetro de 16mm y el vástago de 6mm; la presión del aire comprimido es de 10 bar(1 N/mm^2) y el rendimiento del 80%.

$$F_{ea} = \eta * P * (\pi * D^2/4)$$
$$F_{er} = \eta * P * [\pi * (D^2 - d^2)/4]$$

D = Diametro del émbolo (mm)
d = Diametro del vástago (mm)
η = Eficiencias
P = Presión

CilindoDobleEfecto.java

Sentencia de decisión IF-ELSE

```
if(condicionVerdadera){
    //en caso que sea verdadero, ejecuta éste código
}else{
    //en caso contrario, se ejecuta éste código
}
```

Ejercicio - Qué sexo eres

Generar un programa que pregunte qué sexo eres, si es Hombre que diga “Macho alfa lomo plateado”, de lo contrario que diga “Eres una linda señorita”.

Sentencia de decisión SWITCH

```
switch(variable){
    case opcion1:
    //código
    break;
    case opcion2:
    //código
    break;
    default:
    //código
}
```

Opción a comparar

Caso

Rompe el switch

Ejecuta ésta sección si no se cumple ningún caso

Ejercicio - Ley de Ohm (switch)

Crear un menú dando las opciones para seleccionar que se desea calcular. Opciones: 1. Voltaje, 2. Corriente, 3 Resistencia y al final arrojar el resultado de la operación.

LeyOhmSwitch.java

Sentencias de control - FOR

Separado por punto y coma (;)

```
for(inicio; condicion; Δ){

    //código que se va a repetir hasta que la condición sea falsa

}
```

Ejercicio - Imprimiendo tablas de multiplicar

- Realizar un programa que imprima la tabla de 7, que llegue hasta el 10. utilizando la estructura de control for.
- Realizar un programa que realice la tabla que el usuario quiera conocer, debe llegar hasta el 10.

TablaFor.java

Ejercicio - Estadística

- Solicitar al usuario la cantidad de números que va a ingresar y calcular la media.
- Pedir a usuario, cada número de un conjunto y dar el resultado de la media.

$$\bar{X} = \frac{\sum_{i=1}^{n} x_i}{n}$$

Estadistica.java

For infinito - break

```
for( ; ; ){//no lleva argumentos pero si los punto y coma(;)
    //cuerpo del ciclo
    if(codicion){//si cumple con la condición rompe el for
        break; //rompe el ciclo
    }
}
```

Ejercicio - Resistencia en serie

- Realizar un programa que calcule la resistencia equivalente en serie. Se le preguntará al usuario cuantas resistencias tiene, posterior, irá solicitando una a una.
- Realizar un programa que calcule la resistencia equivalente en serie. El usuario puede dar n resistencias, y cuando el usuario le el valor de -1, el programa hará el cálculo, dando el resultado.

ResistenciaSerie.java

Sentencias de control - WHILE & DO-WHILE

```
while(condicionVerdadera){

    //código que se ejecuta
mientras la condición de True

}
```

```
do{

    //código a ejecutar mientras la
condición sea True

}while(condicionVerdadera);
```

Ejercicio - Imprimiendo tablas de multiplicar (while)

- Realizar un programa que imprima la tabla de 9, que llegue hasta el 10. utilizando la estructura de control for.
- Realizar un programa que realice la tabla que el usuario quiera conocer, debe llegar hasta el 10.

TablaWhile.java

Arreglos (array)

Es una estructura de datos, una colección de elementos, en éste caso es una colección de referencias.

Caracteristicas:

- Espacio definido
- Índice de posición
- Solo puede contener un solo tipo elemento

Arreglos (array)

Arreglos (array)

```
tipo nombre[] = new tipo[tamaño];

tipo nombre[] = {valor1, valor2, valor3};
```

```
int miArreglo[] = new int[4]; //array con 4 espacios

int segundoArreglo[] = {4, 3, 7, 9};//array con 4 espacios
```

Arreglos.java

4. Programación Orientada a Objetos

Objetos

Clase

Objeto

Método y atributos

Objetos

Atributos y Comportamientos

kilometraje

noPuertas

acelerar( );

arrancar( );

color

frenar( );

nivelGasolina

Creación de Objetos

Objeto = Clase

Nombre de clase

```
public class MiObjeto{

    int campo1;

    public void miMetodo(){ //cuerpo del método

    }

}
```

Atributos = campos

Métodos = Comportamientos

Campos y Métodos

```
public int noPuertas; //campo de tipo entero
```

Nivel de acceso

Tipo

Nombre

Parentesis

```
public void acelerar(){ //comienza el método, tipo void

    System.out.println("acelerando");

}
```

Cuerpo del método
(Lo que hará el método)

Creando Objeto

Campos:

- noPuertas: + int
- Kilometraje + long

Comportamientos:

- acelerar(); + void
- arrancar(); + void

Auto.java

Tipos de métodos

Nombre	Descripción
void	No devuelve ningún dato, sólo realiza una acción
int	Devuelve un valor entero
long	Devuelve un valor entero largo
float	Devuelve un valor flotante
double	Devuelve un valor flotante largo
char	Devuelve un valor tipo carácter
byte	Devuelve un valor tipo byte
Tipo de Objeto	Devuelve el tipo del objeto

Niveles de acceso (Encapsulamiento)

Nombre	Clase	Package	Subclase	Todos
public	Sí	Sí	Sí	Sí
protected	Sí	Sí	Sí	No
No especificado	Sí	Sí	No	No
private	Sí	No	No	No

Crear objeto - Resistencia

Campos:

- `valorOhmico: + int`
- `potencia: + double`

Métodos:

- `resistiendo(): + void`

ResistenciaBasica.java

Métodos con *argumentos*

Nivel de acceso

Tipo

Nombre del método

Argumentos separados por coma (,)

```
public double resistencia(double voltaje, double corriente){

    double resultado = voltaje/corriente;

    return resultado;

}
```

Cuerpo del método

Valor que retorna, regresa a quien lo llamó

Objeto - Ley de Ohm

Métodos:

- resistencia(): + double
- corriente() : + double
- voltaje() : + double

LeyOhm.java

Objeto - Cálculo de Resistencias

Métodos:

- serie(double[] res): + double
- paralelo(double[] res): + double
- calculoSerie(): - double
- calculoParalelo(): - double

CalculoResistencia.java

Alcance de variables

```
public class MiClase{

    public int variable1;//variable global

    public void metodo1(){

        int variable2;  //variable local

        for(int variable3 = 0; variable < 6; variable++){

            //cuerpo del for

        }

    }

}
```

Sobrecarga de métodos

```
public double resistenciaSerie(double resistencia1, double
resistencia2){

    //código

}

public double resistenciaSerie(double resistencias[]){

    //código

}
```

ResistenciaSerie.java

Argumentos de longitud variable

```
public double resistenciaSerie(double resistencia...){

    //código

}
```

CalculoResistencia.java

Constructores

```
public class MiObjeto{
```

Nivel de acceso

Mismo nombre de la clase

```
    public MiObjeto(){ //constructor por default

        //contenido del constructor

    }
```

- El lo primero que se manda a llamar al crear una instancia
- Debe tener el mismo nombre que la clase

```
}
```

Constructores con argumentos

```
public class MiObjeto{

    public MiObjeto(){ //constructor vacío

    }

    public MiObjeto(tipo argumento){ //constructor con argumento

    }

}
```

Constructores2.java

Metodos estáticos

```
public static void metodoEstatico(){

    //cuerpo del método

}
```

- Se requiere que se genere una **instancia**
- No se crea uno nuevo aunque se genere una instancia

DesviacionEstandar.java;AreaCirculo.java

Campos FINAL

Una variable tipo FINAL significa que ese valor jamás va a cambiar, no se puede cambiar una vez dado un valor.

La forma de declararlo es en Mayúsculas y separados por guión bajo (_)

```
final double PI = 3.141592653589793;
```

```
final long VELOCIDAD_DE_LA_LUZ = 300000000;
```

Manejo de Excepciones (Básico)

```
try{
    //ejecución de código normal
}catch(Exception){
    //si sucede un error, se trata el error aquí
}
```

5. Interfaz Gráfica

Elementos de Interfaz - Ventana

Elementos de Interfaz

Botones(Button)

Etiquetas (Label)

Elementos de Interfaz

Campo de texto

Área de texto

Elementos de Interfaz

Botón tipo radio

RadioButton

Caja de chequeo

CheckBox

Eventos

Un evento es un método que se dispara cuando sucede un cambio en un elemento.

Eventos:

- *Click* (Action)
- *Mouse*
  - Entrar el mouse
  - Sale el mouse
- *Teclado*
  - Teclas
- …

Ejercicios con Interfaz

- Ley de Ohms
- Neumática de pistón de doble efecto y efecto simple
- Calculo de corriente del secundario y primario de un transformador monofásico, conociendo su impedancia del secundario

Interfaz - Ley de Ohms

Descargar archivos

---

https://github.com/jalmx89/Curso-CBTIS-85

Contacto

Ing. Alejandro Leyva
http://www.alejandro-leyva.com
contacto@alejandro-leyva.com
https://www.linkedin.com/in/alejandro-leyva-consult/
https://www.facebook.com/leyva.consult/