UNIVERSIDAD AUTÓNOMA DE SINALOA

Facultad de Informática Culiacán

Introducción al Lenguajes Java

Instructor:

MC. Gerardo Gálvez Gámez

gerardo.galvez@uas.edu.mx

Noviembre de 2016

¿Qué es Java?

- Java es la base para prácticamente todos los tipos de aplicaciones de red, además del estándar global para desarrollar y distribuir aplicaciones móviles y embebidas, juegos, contenido basado en web y software de empresa.
- Java es un lenguaje de programación orientado a objetos desarrollado a principios de los años 90.
- Su sintaxis es tomada de C y C++, con un modelo de objetos más simple, eliminando la herramientas de bajo nivel.
- Es independiente de la plataforma. Las primeras implementaciones de Java rezaban:*"write once, run anywhere".*

Breve historia …

- Java se creó originalmente como una herramienta de programación para un proyecto set-top-box conocido como *7.

- Fue realizado por un equipo de 13 personas, dirigidas por James Gosling.
- Los objetivos de Gosling eran implementar una máquina virtual y un lenguaje con una estructura y sintaxis similar a C++.
- Nace como un lenguaje ideado en sus comienzos para programar electrodomésticos en la compañía de Sun MicroSystems

Introducción al Lenguaje Java• FIUAS

Breve historia …

- En un principio, el sistema *7 no encontró un lugar en el mercado.
- A principios de los noventa, y sin un mercado para su herramienta, Gosling y su equipo se reunieron y notaron que “la nueva y popular Internet” tenia exactamente el tipo de configuración de red que ellos habían visionado para la industria de la TV por cable.

- Con esto en mente Gosling y su equipo crearon el navegador WebRunner y realizaron un demo que mostraba una molécula animada en una reunión de profesionales de la industria del entretenimiento e Internet.

Breve Historia

- El lenguaje se denominó inicialmente **"Oak"**. Luego pasó a denominarse **"Green"** tras descubrir que Oak era ya una marca comercial registrada.
- El término **"JAVA"** fue acuñado en una cafetería frecuentada por algunos de los miembros del equipo.
- No está claro si es un acrónimo o no, algunas hipotesis indican que podría tratarse de las iniciales de sus creadores: **J**ames Gosling, **A**rthur **V**an Hoff, y **A**ndy Bechtolsheim. Otras abogan por "**J**ust **A**nother **V**ague **A**cronym".
- La hipótesis que más fuerza tiene es la que Java debe su nombre a un tipo de café disponible en la cafetería.

Introducción al Lenguaje Java• FIUAS

Características de Java

- **Lenguaje Simple.-**Basado en el lenguaje C y C++ pero elimina características que se utilizan esporádicamente y que creaban problemas a los programadores.
- **Orientado a Objetos.-brinda** soporte a las técnicas de desarrollo POO y a la reutilización de componentes de software
- **Solido.-No s**e bloquea fácilmente ante errores de programación, ya que no permite realizar operaciones que corrompan el código.
- **Seguro.-** Evita la corrupción de código
- **Multihilos.- Se** aplica a la realización de aplicaciones en las que ocurra más de una cosa a la vez.
- **Dinámico.-** No exige se compile de nuevo la aplicación al cambiar una clase madre.
- **Gestión de Memoria.-** Posee un sistema de administración de memoria automático.
- **Multiplataforma.-** Con el cual se pueden desarrollar programas que se ejecuten sin problemas en sistemas operativos como Windows, Linux, Mac, Unix, etc

Aplicaciones que se pueden crear con Java

- **Aplicaciones** – Programas tradicionales que se ejecutan en la computadora.
  - Aplicaciones de red,
  - Aplicaciones móviles y embebidas,
  - Juegos,
  - Contenido basado en web
    - ***Applets*** – pequeños programas que se ejecutan dentro de una página Web.
  - Software de empresa (Inventarios, Facturación, etc.)

Algo más

- 1**. Aplicaciones "cliente"**: son las que se ejecutan en una sola computadora(por ejemplo el portátil de tu casa) sin necesidad de conectarse a otra máquina. Pueden servirte por ejemplo para realizar cálculos o gestionar datos.
- 2**. Aplicaciones "cliente/servidor"**: son programas que necesitan conectarse a otra máquina (por ejemplo un servidor de datos) para pedirle algún servicio de forma más o menos continua, como podría ser el uso de una base de datos. Pueden servir por ejemplo para el teletrabajo: trabajar desde casa pero conectados a una computadora de una empresa.
- 3. **Podemos hablar también de "aplicaciones web"**, que son programas Java que se ejecutan en un servidor de páginas web. Estas aplicaciones reciben "solicitudes" desde una computadora y envían al navegador (Internet Explorer, Firefox, Safari, etc.) que actúa como su cliente páginas de respuesta en HTML.

Aplicaciones que se pueden crear con Java

- El 97% de los escritorios empresariales ejecutan Java
- El 89% de los escritorios (o computadoras) en Estados Unidos ejecutan Java
- 9 millones de desarrolladores de Java en todo el mundo
- La primera opción para los desarrolladores
- La primera plataforma de desarrollo
- 3 mil millones de teléfonos móviles ejecutan Java
- El 100% de los reproductores de Blu-ray incluyen Java
- 5 mil millones de Java Cards en uso (sim en teléfonos, Monedero electrónico)
- 125 millones de dispositivos de televisión ejecutan Java
- 5 de los 5 principales fabricantes de equipos originales utilizan Java ME

https://www.java.com/es/about/

Versiones de java

JDK 1.0 (1996): primer lanzamiento del lenguaje Java.

· **JDK 1.1** (1997): mejora de la versión anterior.

· **J2SE 1.2** (1998): ésta y las siguientes versiones fueron recogidas bajo la denominación Java 2 y el nombre "J2SE" (Java 2 Platform, Standard Edition), reemplazó a JDK para distinguir la plataforma base de J2EE (Java 2 Platform, Enterprise Edition) y J2ME (Java 2 Platform, Micro Edition). Incluyó distintas mejoras.

· **J2SE 1.3** (2000): mejora de la versión anterior.

· **J2SE 1.4** (2002): mejora de la versión anterior.

· **J2SE 5.0** (2004): originalmente numerada 1.5, esta notación aún es usada en ocasiones. Mejora de la versión anterior.

· **Java SE 6** (2006): en esta versión, Sun cambió el nombre "J2SE" por Java SE y eliminó el ".0" del número de versión. Mejora de la versión anterior.

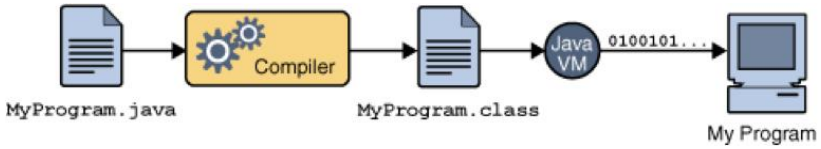
· **Java SE 7** (2011): nueva versión que mejora la anterior.

· **Java SE 8 Update 111 y 112 (18 de octubre de 2016)**

Java SE 9: nueva versión que mejora la anterior (en difusión).

·**Java SE 10:** nueva versión que mejora la anterior (todavía sin uso comercial).

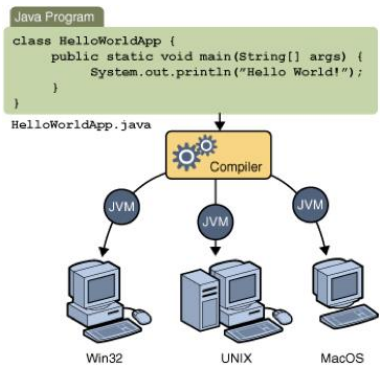
.java, .class, JVM???

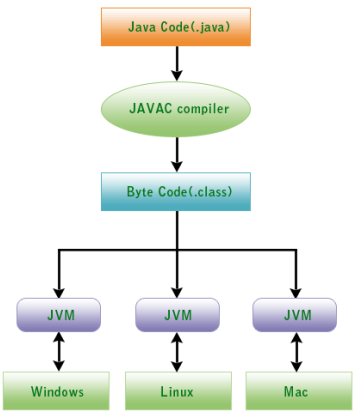


- En Java el código fuente se escribe en un archivo de texto plano con extensión **.java**.
- Luego, el código es compilado a archivos **.class**. Un archivo .class no contiene código nativo a un tipo de procesador, en cambio contiene **bytecodes**.
- Finalmente, la aplicación es interpretada por la máquina virtual de Java, transformando los bytecodes en código nativo en tiempo de ejecución.

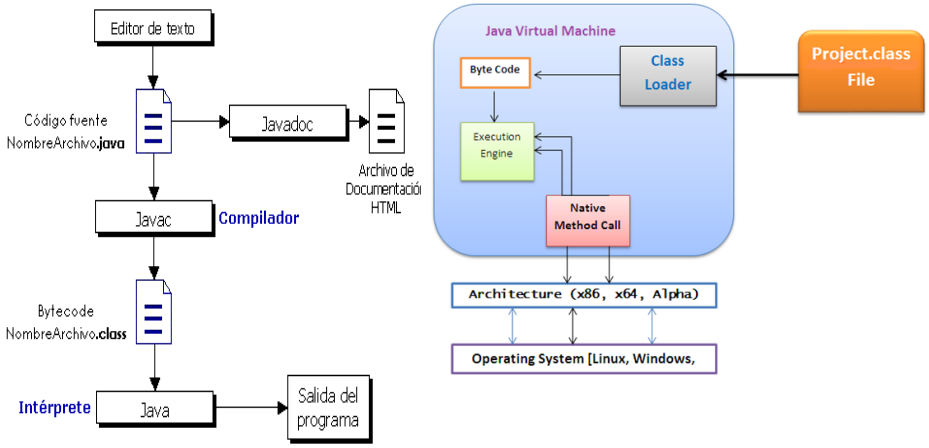
Bytecode

- El bytecode es lenguaje nativo de cualquier implementación de la maquina virtual de Java. De esta forma se logra que un programa Java corra en cualquier plataforma que disponga de una JVM(*Java Virtual Machine*).



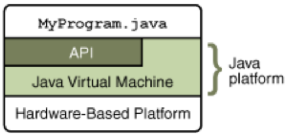


Compilación y ejecución del programa



La plataforma Java

- Una plataforma es el ambiente de software o hardware en el que corre un programa.
- La plataforma Java consta de dos componentes
  - La maquina virtual de Java
  - La API de Java (*Application Programming Interface*)



- La API de Java es una vasta colección de componentes de software que proveen un conjunto de funciones útiles.

Compilación y ejecución del programa

Para compilar y ejecutar un programa desde el *command prompt* es necesario hacer tres cosas:

1. Indicarle al sistema operativo dónde está el compilador y el JVM:

   ```
   set path=C:\Program Files\Java\jdk1.6.0_07\bin
   ```

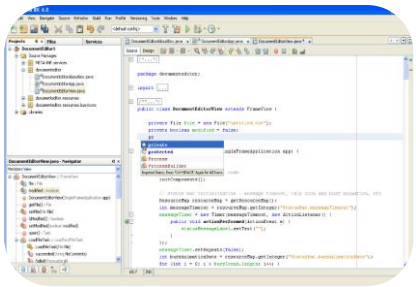
2. Compilar el programa:

   ```
   javac HelloWorldApp.java
   ```

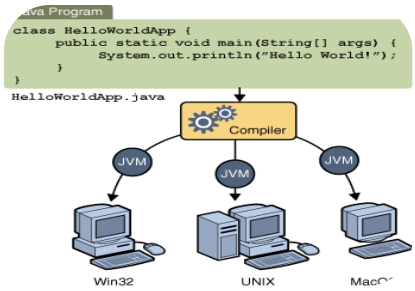
3. Invocar el JVM para ejecutar el programa:

   ```
   java HelloWorldApp
   ```


Herramientas de programación en Java



Entornos de desarrollo



Compilador

Java Development Kit (JDK)

- **El JDK consiste de lo siguiente:**
  - El Java development tools, incluyendo el compilador, debugger y el intérprete Java.
  - Las Java class libraries organizadas como una colección de paquetes.
  - Un número importante de programas de demostración.
  - Varias herramientas de soporte y componentes, incluyendo el código fuente de las clases en la biblioteca.
- URL*: java.sun.com/javase/downloads*

Introducción al Lenguaje Java• FIUAS

Ambientes de desarrollo Java (IDEs)

- Un IDE (del inglés Integrated Development Enviroment). Es un ambiente que integra un conjunto de herramientas (editor con prestaciones especiales, compilador, administración de proyectos, debugger, etc) que permiten realizar todo el proceso de desarrollo dentro del mismo.

- Si bien es posible escribir nuestros programas java en cualquier editor de texto y compilarlos desde una consola con *javac*, es mas cómodo y recomendable trabajar desde un IDE.

¿Y que tiene un IDE que no tenga mi bloc de notas?

- En general, todos los editores de los IDEs mas modernos proveen:
  - Un editor ***language aware***: indentación de código, resaltador de sintáxis, matching de variables y llaves, etc.
  - ***Parsing en tiempo de ejecución***: corrección de errores, remarcado de ocurrencias, tips, arreglos sencillos, etc.
  - Generación automática de código y funciones para completar código mientras escribimos.
- Administración de proyectos
- Debugger
- Múltiples opciones de configuración

Introducción al Lenguaje Java• FIUAS

Algunos de los IDEs mas utilizados

www.eclipse.org

www.bluej.org

www.netbeans.org

www.jetbrains.com/idea

- Es un ambiente de desarrollo de código abierto.
- Originalmente desarrollado por ***Sun Microsystems***, actualmente mantenido por la comunidad ***NetBeans*** *(aunque aun recibe soporte de Sun como producto)*.
- Es ***multiplataforma***
- Licencias ***CDDL(***(Licencia Común de Desarrollo y Distribución)***) y GPL 2 (***Licencia Pública General ***)***.
- Todas las funciones en NetBeans son provistas por ***módulos***.

Introducción al Lenguaje Java• FIUAS

Proceso de instalación de NetBeans

- **Bajar e instalar la última versión del JDK *(requerido)*.**

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

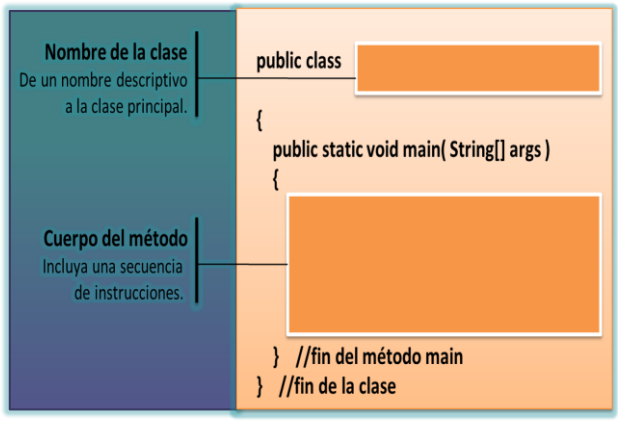
- **Descarga del IDE**
  - ***https://netbeans.org/downloads/***

  y bajar la versión correspondiente a nuestro sistema operativo.
- Ejecutar el instalador y seguir el proceso de instalación y ejecutar la aplicación ***netbeans*** al finalizar el mismo.

Estructura de un programa Java

- La programación orientada a objetos se basa en que cada programa es ***una simulación de un mundo real o virtual***.
- Cada uno de estos mundos esta compuesto de ***objetos***.
- Los objetos se comunican a través de ***mensajes***
- Por lo tanto, un programa orientado a objetos no es mas que ***una configuración de un conjunto de objetos y los mensajes que se envían entre ellos***.
- El *"molde"* que guarda la descripción de todos los objetos de un mismo tipo e lo que conocemos como ***clase***.
- A su vez, las clases con propósitos similares pueden agruparse en ***paquetes***.
- La ejecución de un programa comienza en el método ***"main"*** de una clase.

ESTRUCTURA



Donde Nombre de la clase es el nombre de la clase principal que contiene el código fuente que deberá guardarse en la computadora con el sufijo .java (NombreClase.java).

Todas las aplicaciones Java tienen un método main que a su vez, contiene un conjunto de instrucciones.

En Java los conjuntos o bloques de sentencias se indican entre llaves ({ y }).

Estructura de un programa

```
package persona;

import java.util.Vector;

public class Persona {

    private Vector companieros;

    public Persona() {
        this.companieros = new Vector();
    }

    public Persona(Vector amigos) {
        this.companieros = amigos;
    }

    public void addCompaniero(Persona unaPersona){
        this.companieros.add(unaPersona);
    }

    public Persona getPrimerCompaniero(){
        return (Persona)this.companieros.remove(0);
    }

}
```

Paquete

Imports

Declaración de la clase

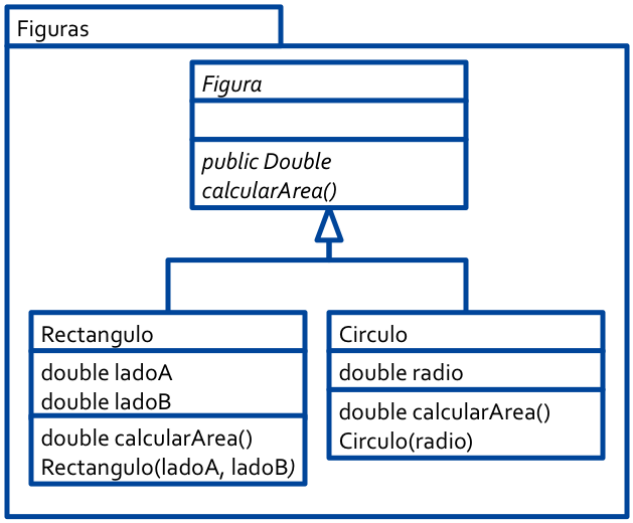
Variables

Constructores

Métodos

Introducción al Lenguaje Java• FIUAS

Ejemplo de paquete



Reglas para la creación de clases y paquetes

- Las clases Java se implementan en archivos separados.
- Cada clase se debe encontrar dentro de un paquete.
- El nombre del archivo de la implementación de clase debe ser igual al nombre de la clase.
- La estructura de paquete debe corresponderse a la estructura de directorio. Ej: edu.sun. Ejemplo seria mapeado al archivo .../edu/sun/Ejemplo.java

Instrucciones de Salida

Aplicaciones consola:

La sintaxis básica es:

```
System.out.println("Mensaje a mostrar");
```

Ejemplo:

```
System.out.println ("El precio es de " + precio + " Pesos");
```

Instrucciones de Entrada

- La entrada de datos por el teclado (console input) no está directamente incluida en JAVA. Se puede utilizar la clase Scanner para crear un objeto que "lea" el dato desde System.in, como se muestra a continuación:

```
Scanner input = new Scanner(System.in);
```

Declara una variable de input cuyo tipo es de Scanner

Crea un objeto del tipo Scanner

La instrucción completa crea un objeto del tipo **Scanner** y asigna su referencia a la variable **input**

Ejemplo 1: Lectura de un valor numérico

```
2   //Scanner facilita la lectura de datos y se encuentra en el paquete
3   //java.util (se ocupa importar)
4   import java.util.Scanner;
5   public class Exposicion {
6       public static void main(String[] args){
7   //Se crea un objeto de la clase Scanner
8       Scanner sc = new Scanner(System.in);
9   //Se creo el objeto sc. Hecho esto se puede leer datos por teclado
10  //Declaracion de variables
11      int Numero;
12  //Mandar imprimir para pedir el valor de la variable
13      System.out.print("Introduzca un numero entero: ");
14  //Leer el valor de Numero
15      Numero = sc.nextInt();
16  //Mandar a imprimir el valor obtenido
17      System.out.print("El valor obtenido es: "+Numero);
```

Ejemplo 2:Lectura de un valor alfanumérico

```
Source  History
2   //Scanner facilita la lectura de datos y se encuentra en el paquete
3   //java.util (se ocupa importar)
4   import java.util.Scanner;
5   public class Exposicion2 {
6       public static void main(String[] args){
7   //Se crea un objeto de la clase Scanner
8       Scanner sc = new Scanner(System.in);
9   //Se creo el objeto sc. Hecho esto se puede leer datos por teclado
10  //Declaracion de variables
11      String Nombre;
12  //Mandar imprimir para pedir el valor de la variable
13      System.out.print("Introduzca su nombre: ");
14  //Leer el valor de Nombre
15      Nombre = sc.nextLine();
16  //Mandar a imprimir el valor obtenido
17      System.out.print("Tu nombre es: "+Nombre);
```


Comentarios

- **Comentario de una línea**
  - // Este es un comentario estilo C++, llega al final de la línea
- **Comentario de varias líneas**
  - /* En este otro comentario estilo C,
  - el final lo indica la marca */

CONTROL DE EXCEPCIONES

- Las sentencias que tratan las excepciones son **try y catch.** La sintaxis es la siguiente:

```
try
{
// instrucciones que se ejecutan a menos de que haya un error
}
catch( ClaseExcepcion ObjetoQueCapturaExcepcion)
{
 //instrucciones que tratan la excepción ocurrida
}
```

ClaseExcepcion, es el nombre de la clase que se utilizara para crear el objeto que captura capturara la excepción que se genere, mientras que **ObjetoQueCapturaExcepcion** es el nombre que se le asigno al objeto que se instanció.

Introducción al Lenguaje Java• FIUAS

CONTROL DE EXCEPCIONES

Puede haber más de una sentencia **catch** para un mismo bloque **try.**

Ejemplo:

```
try
{
    readFromFile("arch");
    ...
}
catch(FileNotFoundException e)
{
//archivo no encontrado
...
}
catch (IOException e) {
...
}
```

TIPOS DE DATOS PRIMITIVOS

Son aquellos que no requieren de métodos, debido a que no son objetos; no necesitan una invocación para ser creados

Tipo de dato	Representación	Tamaño (Bytes)	Rango de Valores	Valor por defecto	Clase Asociada
byte	Numérico Entero con signo	1	-128 a 127	0	Byte
short	Numérico Entero con signo	2	-32768 a 32767	0	Short
int	Numérico Entero con signo	4	-2147483648 a 2147483647	0	Integer
long	Numérico Entero con signo	8	-9223372036854775808 a 9223372036854775807	0	Long
float	Numérico en Coma flotante de precisión simple Norma IEEE 754	4	$\pm 3.4x10^{-38}$ a $\pm 3.4x10^{38}$	0.0	Float
double	Numérico en Coma flotante de precisión doble Norma IEEE 754	8	$\pm 1.8x10^{-308}$ a $\pm 1.8x10^{308}$	0.0	Double
char	Carácter Unicode	2	\u0000 a \uFFFF	\u0000	Character
boolean	Dato lógico	-	true ó false	false	Boolean

TIPOS DE DATOS OBJETO

Tipos de la biblioteca estándar de Java	**String** (cadenas de texto) Muchos otros como: (Scanner, TreeSet, ArrayList…)
Tipos definidos por el programador / usuario	Cualquiera que se nos ocurra, por ejemplo Taxi, Autobus, Tranvia
Arrays	Serie de elementos o formación tipo vector o matriz. Lo consideraremos un objeto especial que carece de métodos.
Tipos envoltorio o wrapper (Equivalentes a los tipos primitivos pero como objetos.)	Byte Short Integer Long Float Double Character Boolean

OPERADORES ARITMÉTICOS EN JAVA

OPERADOR	DESCRIPCIÓN
+	Suma
–	Resta
*	Multiplicación
/	División
%	Resto de una división entre enteros (en otros lenguajes denominado mod)

Introducción al Lenguaje Java• FIUAS

FUNCIONES CLASE MATH JAVA

Función matemática	Significado	Ejemplo de uso	Resultado
abs	Valor absoluto	int x = Math.abs(2.3);	x = 2;
atan	Arcotangente	double x = Math.atan(1);	x = 0.78539816339744;
sin	Seno	double x = Math.sin(0.5);	x = 0.4794255386042;
cos	Coseno	double x = Math.cos(0.5);	x = 0.87758256189037;
tan	Tangente	double x = Math.tan(0.5);	x = 0.54630248984379;
exp	Exponenciación neperiana	double x = Math.exp(1);	x = 2.71828182845904;
log	Logaritmo neperiano	double x = Math.log(2.7172);	x = 0.99960193833500;
pow	Potencia	double x = Math.pow(2.3);	x = 8.0;
round	Redondeo	double x = Math.round(2.5);	x = 3;
random	Número aleatorio	double x = Math.ramdom();	x = 0.20614522323378;

http://docs.oracle.com/javase/8/docs/api/java/lang/Math.html

OPERADORES LÓGICOS PRINCIPALES EN JAVA

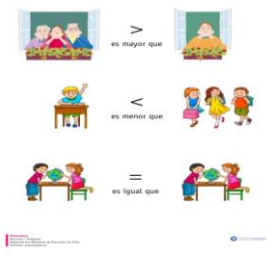
OPERADOR	DESCRIPCIÓN
==	Es igual
!=	Es distinto
<, <=, >, >=	Menor, menor o igual, mayor, mayor o igual
&&	Operador and (y)
||	Operador or (o)
!	Operador not (no)

Introducción al Lenguaje Java• FIUAS

OPERADORES DE ASIGNACIÓN

Se utilizan para asignar un valor nuevo a una variable, propiedad, evento o elemento de indizador.

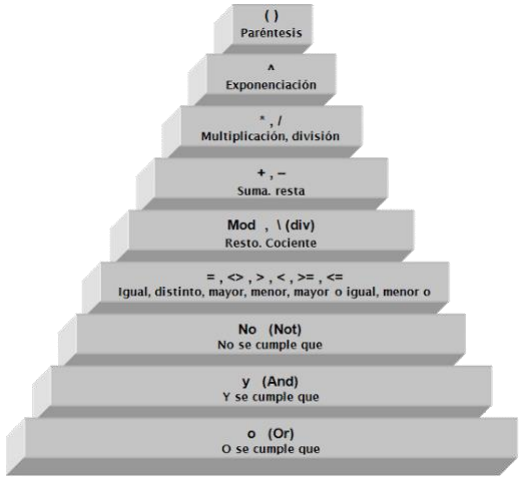
Operadores de asignación		
=	Asignación	a = b
+=	Suma y asignación	a += b (a=a + b)
-=	Resta y asignación	a -= b (a=a - b)
*=	Multiplicación y asignación	a *= b (a=a * b)
/=	División y asignación	a / b (a=a / b)
%=	Módulo y asignación	a % b (a=a % b)



OPERADORES ESPECIALES

Operadores especiales		
++	Incremento	a++ (pos incremento) ++a (pre incremento)
--	Decremento	a-- (postdecremento) --a (predecremento)
(tipo)expr	Cast	a = (int) b
+	Concatenación de cadenas	a = "cad1" + "cad2"
.	Acceso a variables y métodos	a = obj.var1
( )	Agrupación de expresiones	a = (a + b) * c

ORDEN DE PRIORIDAD, PRELACIÓN O PRECEDENCIA



PALABRAS RESERVADAS

Las palabras reservadas se pueden clasificar en las siguientes categorías:

- Tipos de datos: **boolean, float, double, int, char**
- Sentencias condicionales: **if, else, switch**
- Sentencias iterativas: **for, do, while, continue**
- Tratamiento de las excepciones: **try, catch, finally, throw**
- Estructura de datos: **class, interface, implements, extends**
- Modificadores y control de acceso: **public, private, protected, transient**
- Otras: **super**, **null, this**.

Introducción al Lenguaje Java• FIUAS

REGLAS Y RECOMENDACIONES PARA FORMAR IDENTIFICADORES

Para formar un Identificador, debemos considerar las siguientes reglas:

- Todos los identificadores deben de comenzar con una letra, el carácter subrayado (_) o el carácter dólar ($).
- Puede incluir, pero no comenzar por un número.
- No puede incluir el carácter espacio en blanco.
- Distingue entre letras mayúsculas y minúsculas. Para Java una letra en mayúscula es diferente a dicha letra en minúscula, por ejemplo, el identificador numero es diferente al identificador NUMERO.
- No se pueden utilizar las palabras reservadas como identificadores.

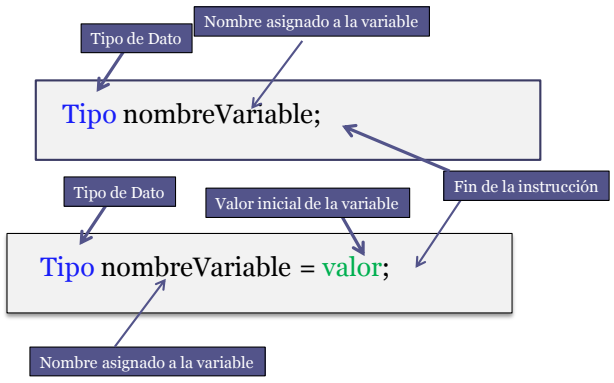
REGLAS Y RECOMENDACIONES PARA FORMAR IDENTIFICADORES

Tipo de identificador	Convención	Ejemplo
Nombre de una clase	Comienza por letra mayúscula	String, Rectangulo, CinematicaApplet
Nombre de función	comienza con letra minúscula	calcularArea, getValue, setColor
Nombre de variable	comienza por letra minúscula	área, color, appletSize
Nombre de constante	En letras mayúsculas	PI, MAX_ANCHO

Introducción al Lenguaje Java• FIUAS

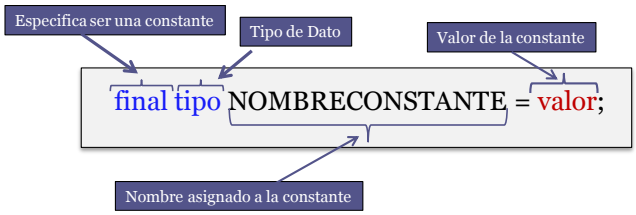
VARIABLES

- **Sintaxis:**



CONSTANTES

- Para crear una constante se utiliza la palabra reservada ***final***, la cual indica que no puede ser modificado el valor asignado cuando se ha creado.
- **Sintaxis:**

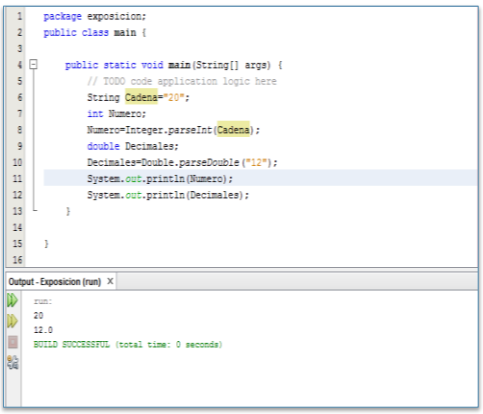


CONVERSIÓN DE TIPOS DE DATOS 1/2

- La conversión de tipos consiste en pasar un tipo de dato a otro para poder realizar cierto uso de este según lo necesitemos.

```
String Cadena="20";
int Numero;
Numero=Integer.parseInt(Cadena);
```

- Como se observa estamos convirtiendo la Cadena “20” a un numero Entero, se usa la clase Integer y el método parseInt para los números enteros



```
package exposicion;
public class main {

    public static void main(String[] args) {
        // TODO code application logic here
        String Cadena="20";
        int Numero;
        Numero=Integer.parseInt(Cadena);
        double Decimales;
        Decimales=Double.parseDouble("12");
        System.out.println(Numero);
        System.out.println(Decimales);
    }

}
```

```
Output - Exposicion (run)
run:
20
12.0
BUILD SUCCESSFUL (total time: 0 seconds)
```

CONVERSIÓN DE TIPOS DE DATOS 2/2

- No todos los tipos se convertirán de forma segura. Por ejemplo, al convertir un *long* en un *int*, el compilador corta los 32 bits superiores del *long* (de 64 bits), de forma que encajen en los 32 bits del *int*, con lo que si contienen información útil, esta se perderá.
- Por ello se establece la norma de que "en las conversiones el tipo destino siempre debe ser igual o mayor que el tipo fuente"

Tipo Origen	Tipo Destino
Byte	Double, Float, Long, Int, Char, Short
Short	Double, Float, Long, Int
Char	Double, Float, Long, Int
Int	Double, Float, Long
Long	Double, Float
Float	Double
*Los tipos de datos String no necesitan conversión.	

Introducción al Lenguaje Java• FIUAS

MÉTODO VALUEOF PARA CONVERSIÓN DE TIPOS

El uso típico de valueOf es para convertir tipos primitivos en objetos

EXPRESIÓN	INTERPRETACIÓN aprenderaprogramar.com
miInteger = miInteger.valueOf (i)	Con i entero primitivo que se transforma en Integer
miInteger = miInteger.valueOf (miString)	El valor del String se transforma en Integer
miString = miString.valueOf (miBooleano)	El booleano se transforma en String “true” o “false”
miString = miString.valueOf (miChar)	El carácter (char) se transforma en String
miString = miString.valueOf (miDouble)	El double se transforma en String. Igualmente aplicable a float, int, long.

La Clase String.format

```
public class StringFromatExample {
   public static void main(String[] args) {
      System.out.printf("Floating point number with 3 decimal digits: %.3f\n",1.21312939123);
      System.out.printf("Floating point number with 8 decimal digits: %.8f\n",1.21312939123);
      System.out.printf("String: %s, integer: %d, float: %.6f", "Hello World",89,9.231435);

      //definiendo una variable cadena
      String S= String.format(("Floating point number with 3 decimal digits: %.3f\n",1.21312939123);
      System.out.printf(S);

      }
}
```

Salida:

Integer : 15 Floating point number with 3 decimal digits: 1.213
Floating point number with 8 decimal digits: 1.21312939
String: Hello World, integer: 89, float: 9.231435

Introducción al Lenguaje Java• FIUAS

String.format

Conversor	Valor
%b	Booleano
%h	Hashcode
%s	Cadena
%c	Caracter unicode
%d	Entero decimal
%o	Entero octal
%x	Entero hexadecimal
%f	Real decimal
%e	Real notación científica
%g	Real notación científica o decimal
%a	Real hexadecimal con mantisa y exponente
%t	Fecha u hora

http://docs.oracle.com/javase/1.5.0/docs/api/java/util/Formatter.html#syntax

Conversión a formato Moneda

Start Page | PrimerEjemplo.java

Source | History

```
6   package primerejemplo;
7
8   import java.text.NumberFormat;
9   import java.util.Scanner;
10  public class PrimerEjemplo {
11      public static void main(String[] args) {
12          // TODO code application logic here
13          Scanner Leer=new Scanner(System.in);
14          NumberFormat FormatoMoneda=NumberFormat.getCurrencyInstance();
15          float Sueldo,Pi=3.1416f;
16
17          System.out.print("Cual es tu sueldo:");
18          Sueldo=Leer.nextFloat();
19         //salidas
20         System.out.printf("Tu Sueldo es :%" + FormatoMoneda.format(Sueldo));
21      }
22
23  }
```


Codificación de Algoritmos en Pseudocódigos con Estructura Secuencial al Lenguaje.

Pseudocódigo

Objetivo: Calcular la Superficie de una Circunferencia

```
INICIO
    CONST REAL  PI = 3.1416
    ENTERO  Radio, Superficie
    LEER  Radio
    Superficie = PI*Radio*Radio
    IMPRIMIR "La superficie de la
    circunferencia de radio ", Radio, "es igual a ",
    Superficie
FIN
```


Actividades
ExtraClases

Objetivo
El alumno demostrara la habilidad alcanzada en clases, para codificar pseudocódigos de diversos problemas, que utilizan procedimientos de solución secuenciales.

Preguntas?