Bienvenidos

Universidad
Provincial del Sudoeste
Promoviendo el Desarrollo Armónico de la Región

Clase 09.
Aplicaciones Móviles y Cloud Computing

JAVA

Universidad Provincial del Sudoeste
Promoviendo el Desarrollo Armónico de la Región

Temario

09

Desarrollo de aplicaciones Restfull

- ✓ Websockets
- ✓ Motores de Plantillas
- ✓ Dependencias socket.io / express-handlebars

10

Introducción al lenguaje Java

- ✓ Breve historia de Java
- ✓ Características de JAVA
- ✓ Conceptos básicos y primeros pasos

11

Introducción al desarrollo de apps móviles

- ✓ Conceptos previos
- ✓ Tipos de apps: WPA, nativas, híbridas
- ✓ Introducción a Android Studio

Universidad Provincial del Sudoeste
Promoviendo el Desarrollo Armónico de la Región

Objetivos de la clase

- Conocer las bases del lenguaje JAVA
- Aprender conceptos fundamentales de JAVA y de programación orientada a objetos
- Escribir un programa sencillo en JAVA

WebSockets

Websocket es un protocolo de comunicación basado en TCP para poder establecer esa conexión entre el cliente y el servidor. Esto, como sabemos, es el mismo objetivo que cubre HTTP.

La diferencia sustancial es que websockets permitirá establecer una **comunicación bidireccional** entre el cliente y el servidor. La comunicación bidireccional implica:

- ✓ Que el cliente puede obtener recursos del servidor cuando lo solicite (como en HTTP)
- ✓ Que el servidor pueda entregar información al cliente sin necesidad de que el cliente haga una petición.

Es de suma utilidad en aplicaciones como chats, portales de noticias (que impliquen actualizaciones en tiempo real), juegos en linea, etc.

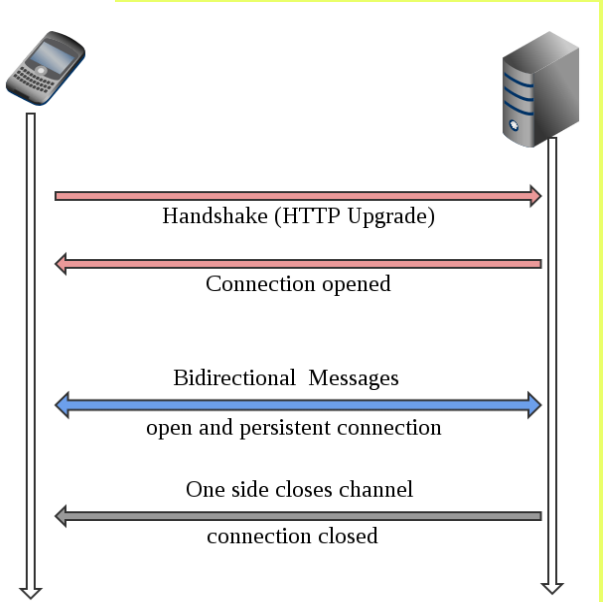
Funcionamiento de un Websocket

Primero, el cliente tiene que enviar una solicitud HTTP llamada **Handshake** (apretón de manos). Este apretón de manos será un "acuerdo" o "contrato" de confianza para que el servidor pueda actualizar al cliente sin que éste se lo pida.

El servidor recibe la petición de Handshake y procede a "responderle el saludo". A esto se le llama "**Abrir conexión**".

A partir de este punto, el canal queda abierto **de manera bidireccional**, por lo que el cliente se puede comunicar con el servidor cuando quiera y viceversa.

La comunicación es "persistente" hasta que alguno de los dos lados decida cerrar el canal de comunicación.



Socket.io

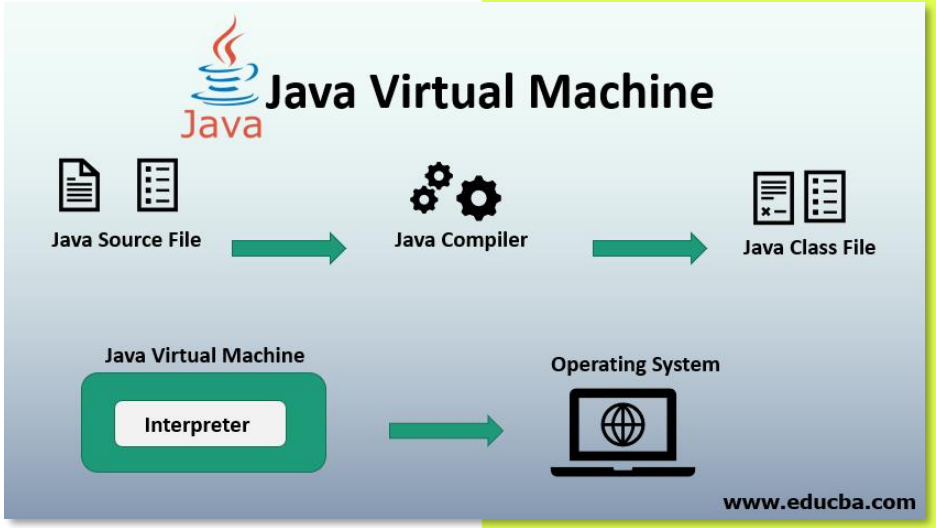
- ✓ Es una biblioteca de Javascript para poder implementar los sockets anteriormente mencionados.
- ✓ Debido al funcionamiento que hemos mencionado antes, **socket.io debe instanciarse tanto del lado del cliente, como del servidor**.
- ✓ Permite utilizar todo el potencial mencionado de los websockets, y cuenta con una API casi idéntica para cliente y para servidor.

Introducción a JAVA

Universidad Provincial del Sudoeste
Promoviendo el Desarrollo Armónico de la Región

JAVA

El lenguaje de programación Java fue desarrollado originalmente por James Gosling, de Sun Microsystems a principios de los ‘90. La primera versión de Java fue publicada en 1995. Sun Microsystems fue adquirida en 2010 por la compañía Oracle.
La sintaxis de Java deriva en gran medida de C y C++, pero tiene menos utilidades de bajo nivel que cualquiera de ellos. Las aplicaciones de Java son compiladas a bytecode (clase Java), que puede ejecutarse en cualquier máquina virtual Java (JVM) sin importar la arquitectura de la computadora subyacente.



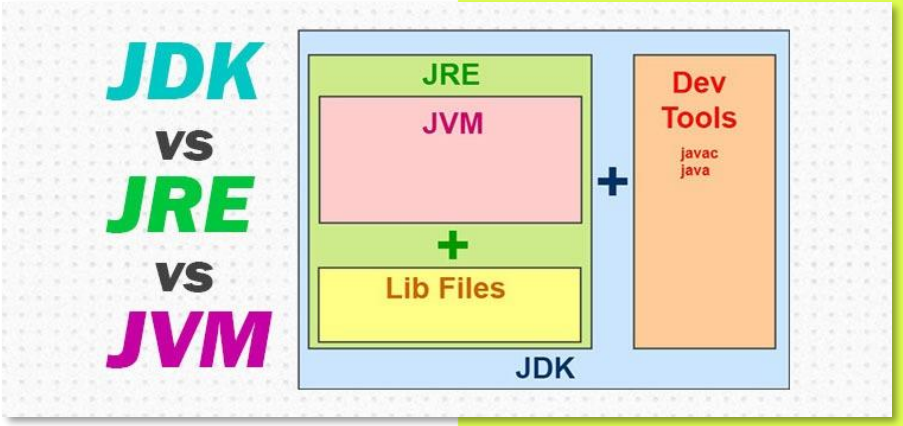
La promesa inicial de los creadores de Java era ***Write Once, Run Anywhere*** (Escríbelo una vez, ejecútalo en cualquier lugar), proporcionando un lenguaje independiente de la plataforma y un entorno de ejecución (la JVM) ligero y gratuito para las plataformas más populares, de forma que los binarios (bytecode) de las aplicaciones Java pudiesen ejecutarse en cualquier plataforma.

JAVA: ediciones / versiones

En la actualidad existen dos versiones, que ofrece Oracle:

- ✓ JSE: Java Standard Edition
- ✓ JEE: Java Entreprise Edition

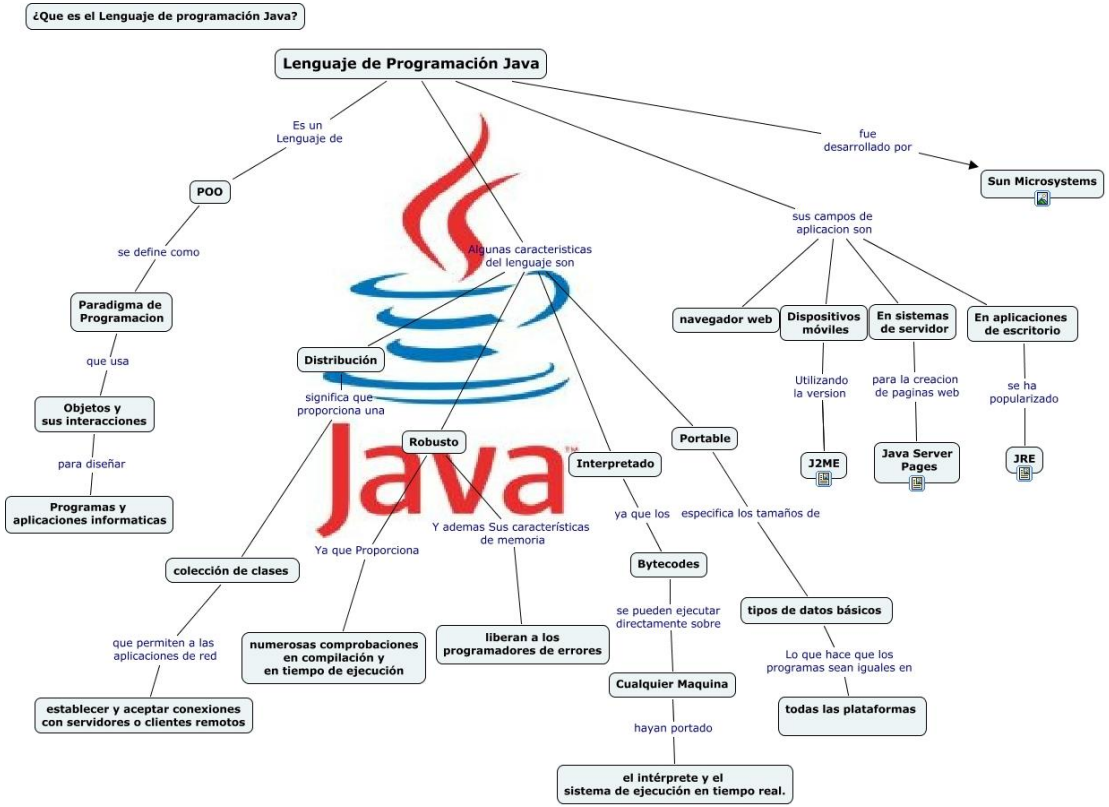
Otro concepto asociado a JAVA es el de JDK, o Java Development Kit: es un software para los desarrolladores de Java. Incluye el intérprete Java, clases Java y herramientas de desarrollo Java. También librerías, y la máquina virtual JVM.



JRE es el entorno de ejecución: contiene algunas librerías, y la JVM.

JVM: es una abstracción. Un conjunto de reglas que debe cumplir un entorno de ejecución para poder ejecutar bytecode. Existe una JVM para cada plataforma: Linux, Windowx, MacOs.

JAVA: características principales



- ✓ Orientado a objetos
- ✓ Multiplataforma
- ✓ Recolector de basura, o garbage collector
- ✓ Fuertemente tipado: cada variable / expresión tiene un Tipo que es conocido en el momento de la compilación. Esto ayuda a detectar errores en etapa de desarrollo.
- ✓ Multitarea
- ✓ Dinámico

Universidad Provincial del Sudoeste
Promoviendo el Desarrollo Armónico de la Región

JAVA: tipos de datos

Listado de tipos de datos ***primitivos*** de JAVA:

Lista de tipos de datos primitivos del lenguaje Java			
Tipo	*Tamaño*	*Valor mínimo*	*Valor máximo*
`byte`	8 bits	-128	127
`short`	16 bits	-32768	32767
`int`	32 bits	-2147483648	2147483647
`long`	64 bits	-9223372036854775808	9223372036854775807
`float`	32 bits	-3.402823e38	3.402823e38
`double`	64 bits	-1.79769313486232e308	1.79769313486232e308
`char`	16 bits	`'\u0000'`	`'\uffff'`

Nota: un dato de tipo carácter se puede escribir entre comillas simples, por ejemplo `'a'`, o también indicando su valor Unicode, por ejemplo `'\u0061'`.

De tipo ***objeto***:

- ✓ Tipos de la biblioteca estándar de Java: String, Scanner, ArrayList, etc.
- ✓ Tipos definidos por el programador: Cliente, Estudiante, Carro, etc.
- ✓ Arrays: elementos tipo vector o matriz.
- ✓ Tipos envoltorio o wrapper: Byte, Short, Integer, Long, Float, Double, Character, Boolean.

Casteo de datos en JAVA

Las variables pueden transformarse en variables de otro tipo de datos. Por ejemplo: si se desea convertir un valor de tipo double a un valor de tipo int se utilizará el siguiente código:

int n;
double x = 82.4;
n = (int) x;

donde la variable n toma el valor 82 (valor en formato entero). El valor de x no se modifica.

La imagen que sigue ilustra algunas de las conversiones que pueden realizarse entre datos de tipo numérico:

```
public class Conversiones {
    public static void main (String [] args) {
        int a = 2;
        double b = 3.0;
        float c = (float) (20000*a/b + 5);
        System.out.println("Valor en formato float: " + c);
        System.out.println("Valor en formato double: " + (double) c);
        System.out.println("Valor en formato byte: " + (byte) c);
        System.out.println("Valor en formato short: " + (short) c);
        System.out.println("Valor en formato int: " + (int) c);
        System.out.println("Valor en formato long: " + (long) c);
    }
}
```

Identificadores en JAVA

Los identificadores son nombres que se les asignan a variables, métodos, clases... en el código fuente de un programa. Los identificadores sólo existen en el código del programa fuente y no en el programa objeto (resultado de la compilación del programa fuente). Todo nuevo identificador que se emplee en un programa Java debe definirse previamente a su utilización. Las normas para la construcción de un identificador empleando el lenguaje de programación Java son las siguientes:

- ✓ Un identificador comienza por una letra, un carácter de subrayado (_) o un carácter de dólar ($). Aunque no se recomienda emplear el carácter $, ya que el compilador suele utilizarlos de forma interna para crear identificadores propios.
- ✓ Los siguientes caracteres pueden ser también dígitos. Pero no pueden emplearse espacios en blanco u otros caracteres como el signo de interrogación (?) o el signo del tanto por ciento (%).
- ✓ No hay límite máximo de caracteres.
- ✓ Se distinguen las mayúsculas de las minúsculas. Por ejemplo, casa, CASA y Casa son tres identificadores diferentes.
- ✓ Existe una serie de palabras reservadas que no pueden emplearse como identificadores

Clases en JAVA

En Java, todo es una clase: al ser un lenguaje orientado a objetos, todo parte de esta estructura de datos.
Una clase es una representación de una entidad de la vida real. Es un molde, a partir del cual podemos crear luego objetos.
La clase cuenta con propiedades y métodos.
Un programa puede construirse empleando varias clases. En el caso más simple se utilizará una única clase. Esta clase contiene el programa, rutina o método principal: main() y en éste se incluyen las sentencias del programa principal. Estas sentencias se separan entre sí por caracteres de punto y coma.

La estructura de un programa simple en Java es la siguiente:

```
public class ClasePrincipal {
    public static void main(String[] args) {
        sentencia_1;
        sentencia_2;
        // ...
        sentencia_N;
    }
}
```

Universidad Provincial del Sudoeste
Promoviendo el Desarrollo Armónico de la Región

Constructores en JAVA

```java
public class Fecha {
    // Atributos o variables miembro
    private int dia;
 private int mes;
    private int anho;

    /**
     * Constructor 1
     * Asigna los valores 1, 1 y 2000 a los atributos
     * dia, mes y anho respectivamente
     */
    public Fecha() {
        this.dia = 1;
        this.mes = 1;
        this.anho = 2000;
    }

    /**
     * Constructor 2
     * @param ndia el dia del mes a almacenar
     * @param nmes el mes del anho a almacenar
     * @param nanho el anho a almacenar
     */
    public Fecha(int dia, int mes, int anho) {
        this.dia = dia;
        this.mes = mes;
        this.anho = anho;
    }

    public String toString() {
        return this.dia + "/" + this.mes + "/" + this.anho;
    }
}
```

Un constructor es un elemento de una clase cuyo identificador coincide con el de la clase correspondiente y que tiene por objetivo obligar a y controlar cómo se inicializa una instancia de una determinada clase, ya que el lenguaje Java no permite que las variables miembro de una nueva instancia queden sin inicializar.
La declaración de un constructor diferente del constructor por defecto, obliga a que se le asigne el mismo identificador que la clase y que no se indique de forma explícita un tipo de valor de retorno. La existencia o no de parámetros es opcional. Por otro lado, la sobrecarga permite que puedan declararse varios constructores (con el mismo identificador que el de la clase), siempre y cuando tengan un tipo y/o número de parámetros distinto.

Arrays en JAVA

Un objeto de la clase predefinida array permite representar una secuencia lineal y finita de elementos del mismo tipo. Estos elementos pueden ser de tipo primitivo o pertenecientes a otras clases.

- ✓ tipoElemento [] identificadorInstancia;
- ✓ int [] numeros;
- ✓ p = new int [5]; // se crea el array de 5 enteros referenciado por p
- ✓ Al primer elemento del array se le asigna el valor -15: p[0] = -15;
- ✓ Al segundo elemento del array se le asigna el valor 26: p[1] = 26;

- ✓ Declaración e inicialización simultanea de arrays: double [] x = {1.5, 2.3, -0.6, 4.8};

```
/**
 * ArrayRaices: Ejemplo de uso de arrays
 */
public class ArrayRaices {
    public static void main (String [] args ) {
        int [] numero = new int[10]; // Array de valores enteros
        double [] raiz = new double[10]; // Array de valores reales

        for (int i=0; i<numero.length; i++) {
            numero[i] = i+1;
            raiz[i] = Math.sqrt(numero[i]);
            System.out.println(numero[i] + " : " + raiz[i]);
        }
    }
}
```

Universidad Provincial del Sudoeste
Promoviendo el Desarrollo Armónico de la Región

Sentencias de control en JAVA

```
System.out.println("Tabla de multiplicar del 5");
for (int i =0 ; i <= 10; i++) {
    System.out.println(5 + " * " + i + " = " + 5 * i);
}
```

```
[inicializacion;]
while (expresionLogica) {
    sentencias;
    [iteracion;]
}
```

```
if (a>b) {
    System.out.println("a es mayor que b");
}  else {
    System.out.println("a no es mayor que b");
}
```

Todo programa o aplicación independiente de Java debe declarar un método principal con la siguiente cabecera:

```
public static void main (String [] args)
```

Packages en Java:

En general, una app de Java se compone de una colección de clases. Cuando los programas son grandes o se trabaja en equipo, es recomendable dividir el código en varios archivos fuente. Las razones son las siguientes: tiempo (compilar solo una parte ante cambios mínimos), eficiencia (para trabajo en equipo, por ej.), y simplicidad para mantener app.
Un package de Java es un conjunto de clases e interfaces relacionadas entre sí.
Por un lado, las clases e interfaces que forman parte de la plataforma Java se estructuran en varios paquetes organizados por funciones o tareas. Por ejemplo, las clases fundamentales están en java.lang, las clases para operaciones de entrada y salida de datos están en java.io, etcétera. Aprender a utilizar y familiarizarse con las clases y métodos implementados en estos paquetes representa la mayor parte del aprendizaje del lenguaje de programación Java.
Por otro lado, cualquier programador puede introducir sus clases e interfaces en paquetes para facilitar tanto su uso en el desarrollo de un programa como su reutilización en varios de ellos. Debe incluir al principio de la clase la siguiente instrucción: ***package identificadordelPaquete;***
Convenciones:

- ✓ Almacenar el código fuente de una clase en un archivo de texto con el mismo identificador que la clase o interfaz y extensión .java. Por ejemplo: Rectangulo.java.
- ✓ Poner el archivo en un directorio con el mismo nombre que el paquete al que pertenezca la clase. Por ejemplo: geometria/Rectangulo.java.

Acceso a packages en Java:

Sólo los componentes públicos de un paquete son accesibles desde fuera del paquete desde el que se han definido. Para emplear un componente público de un paquete debe hacerse una de las siguientes cosas, cada una de las cuales es adecuada a una situación distinta:

✓ Referenciarlo mediante su identificador.

Por ejemplo: geometria.Rectangulo r1 = new geometria.Rectangulo();
Esta opción puede ser adecuada si el identificador sólo se emplea una o muy pocas veces.

✓ Importar el componente del paquete, incluyendo una sentencia import al principio del archivo fuente, antes de cualquier declaración de clase o interfaz y después de la sentencia package, si existe.

```
import geometria.Rectangulo;
...
// Puede hacerse referencia directamente a la clase rectangulo
Rectangulo r1 = new Rectangulo();
```

✓ Importar el paquete completo, empleando la sentencia import con el identificador del paquete seguido de un punto y un asterisco:

```
import geometria.*;
```

Universidad Provincial del Sudoeste
Promoviendo el Desarrollo Armónico de la Región

Documentación Oficial:

Documentación oficial de JAVA, suministrada por Oracle:

- ✓ Documentación oficial Java Release 21

Recursos adicionales:

- ✓ https://www.w3schools.com/java/java_intro.asp
- ✓ https://dev.java/learn/
- ✓ https://roadmap.sh/java

IDE recomendado para JAVA:

- ✓ https://www.jetbrains.com/es-es/idea/

¿Preguntas?

Universidad
Provincial del Sudoeste
Promoviendo el Desarrollo Armónico de la Región

Muchas gracias.

Universidad
Provincial del Sudoeste
Promoviendo el Desarrollo Armónico de la Región