

El Lenguaje y la Plataforma JAVA

La Plataforma abarca dos aspectos:

- Una Plataforma de Software
- Un Lenguaje de Programación

La Plataforma de ejecución JAVA

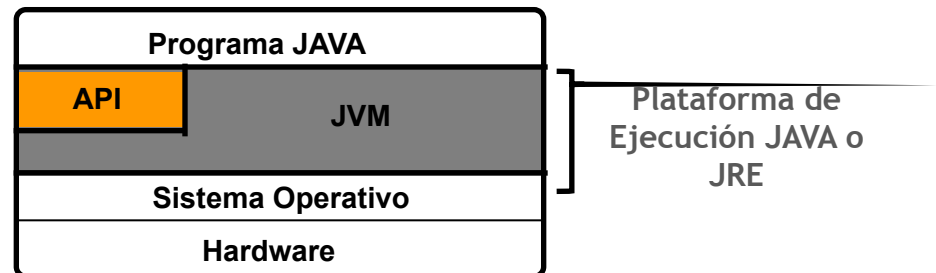
Una **plataforma de ejecución** es la combinación de un hardware y un sistema operativo que provee los servicios necesarios para ejecutar programas. Ejemplos de plataformas de ejecución son: Linux sobre una computadora Intel I7, Solaris ejecutándose sobre hardware Sun, etc.

La **plataforma de ejecución JAVA** provee los servicios necesarios para ejecutar programas escritos en JAVA. La plataforma JAVA se ejecuta sobre cualquier plataforma de ejecución.

A la plataforma de ejecución JAVA se la denomina JRE (Java Runtime Environment) o Entorno de Ejecución JAVA.

La plataforma de ejecución JAVA está compuesta por una **Máquina Virtual JAVA** o **JVM** (Java Virtual Machine) y un conjunto de librerías de código JAVA compilado, comúnmente denominado Interface de Programación de Aplicaciones JAVA o API (del inglés Application Programming Interface). La JVM es parte del JRE.

Los programas escritos en JAVA se ejecutan sobre la plataforma de ejecución JAVA.



Repaso: compilar e interpretar

El **lenguaje de máquina** o **código binario** consiste en instrucciones muy simples que la CPU de la computadora ejecuta directamente. Cada tipo de procesador tiene su propio lenguaje de máquina.

Los programas se escriben en **lenguajes de programación de alto nivel**, como Java, C++, Delphi, Python, etc. Un programa escrito en un lenguaje de alto nivel no puede ejecutarse directamente en una computadora, necesita ser traducido al lenguaje de máquina de la computadora donde se ejecutará. Este proceso de traducción lo realiza un programa llamado **compilador**.

Típicamente, el código ejecutable es particular de la plataforma de ejecución donde se ejecutará el programa; es dependiente de la plataforma de ejecución.

Una alternativa a compilar un programa escrito en un lenguaje de alto nivel, es interpretarlo: un **intérprete** es un programa que traduce y ejecuta un programa escrito en un lenguaje de alto nivel, instrucción por instrucción. A diferencia del compilador, el intérprete no genera un código ejecutable, sino que traduce a código binario y ejecuta, de a una línea por vez, cada vez que se ejecuta el programa.

La Máquina Virtual

La **Máquina Virtual Java** o **JVM** es software y constituye el corazón del JRE.

El lenguaje de máquina de la Máquina Virtual Java es la **codificación de bytes** o **bytecodes**. Es un lenguaje intermedio entre la codificación binaria específica del procesador y el lenguaje de alto nivel.

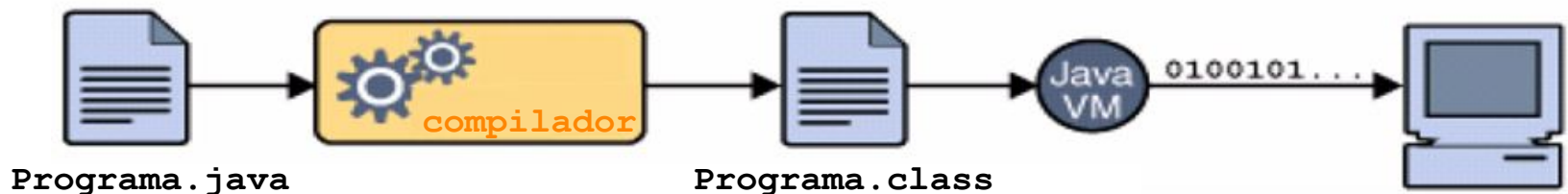
Una de las componentes fundamentales de la JVM es el intérprete JAVA, su responsabilidad es traducir bytecodes a código binario específico y luego ejecutarlo.

Como ya vimos, el código ejecutable es dependiente de la plataforma de ejecución particular -> necesitamos una JVM particular para cada tipo de plataforma de ejecución. Una máquina virtual para Windows, para Linux, para Solaris, etc.

La JVM es un mediador entre los programas JAVA compilados y la plataforma de ejecución específica de la computadora. Sus principales funciones son: traducir y ejecutar bytecodes, administrar la memoria del sistema de ejecución, proveer un sistema de seguridad y balancear la ejecución de múltiples threads.

Los Programas JAVA son de Plataforma Neutral

En JAVA el código fuente tiene extensión *.java*. Los archivos fuentes *.java* se compilan al lenguaje de máquina de la máquina virtual JAVA (JVM). Los archivos compilados tienen extensión *.class*. Un archivo *.class* no contiene código binario para un procesador específico, sino que contiene *bytecodes*, que es la codificación que entiende la máquina virtual de JAVA.



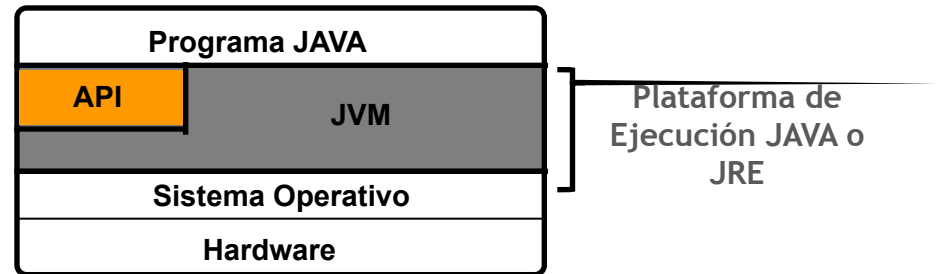
- JAVA es un lenguaje *compilado e interpretado*.
- Para ejecutar un programa JAVA compilado (.class) en una computadora SPARC con Solaris es necesario disponer de la JVM para Solaris, si lo vamos a ejecutar en una computadora Intel con Windows necesitamos la JVM para Windows. El mismo programa JAVA compilado a bytecodes lo podemos ejecutar en diferentes computadoras, no es necesario volver a compilarlo.
- La JVM garantiza que todo programa JAVA es de plataforma neutral.

Más sobre la máquina virtual

Característica principal

- Aislar al programa Java del sistema operativo y del hardware sobre el que se está ejecutando => independencia de la plataforma de ejecución.

Los programas escritos en JAVA se ejecutan sobre la plataforma de ejecución JAVA.

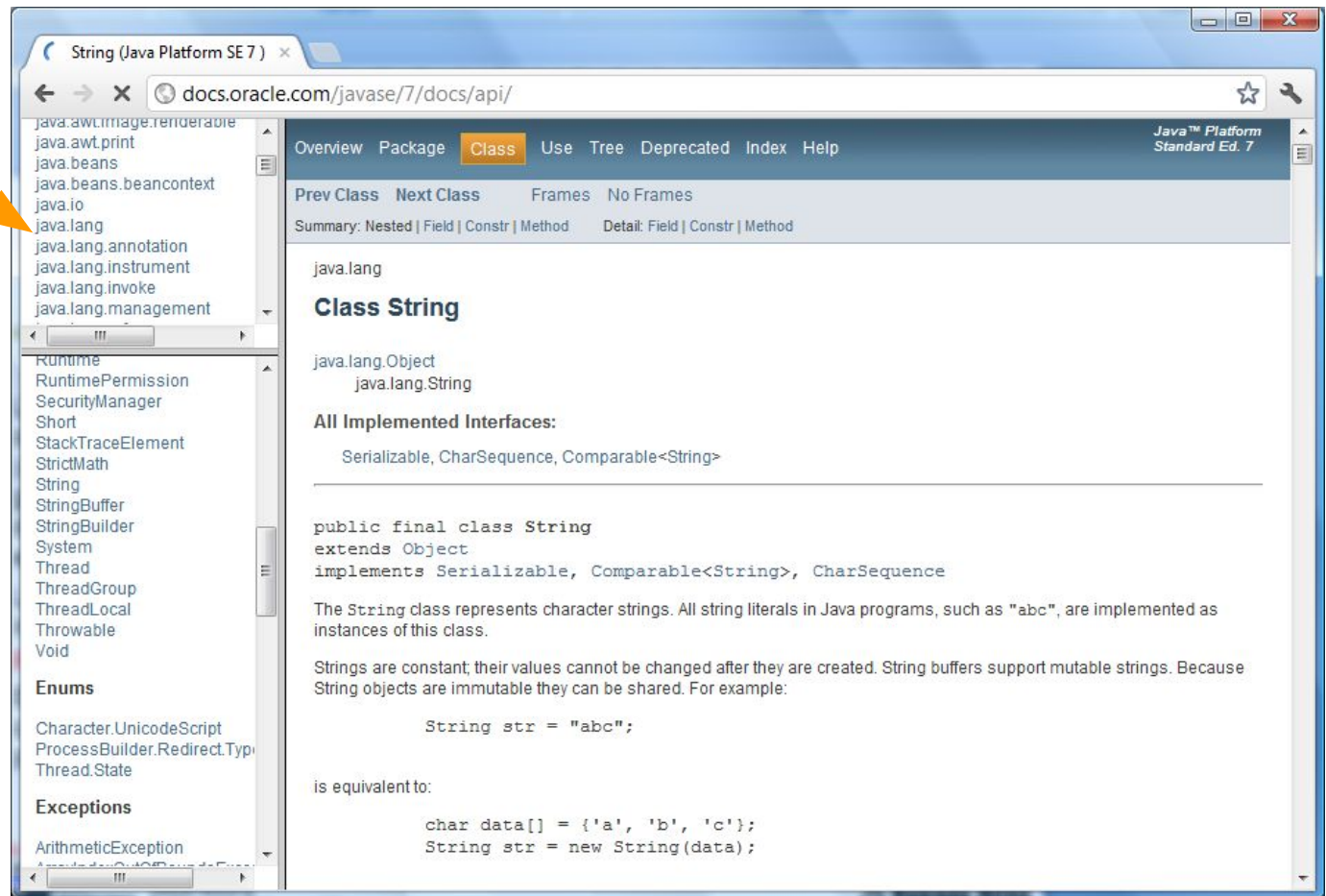


- La especificación de la JVM es única y permite que los programas Java sean independientes de la plataforma de ejecución ya que se compilan para una máquina genérica, la JVM, y se ejecutan en cualquier computadora que disponga de la JVM.
- La JVM asegura la portabilidad de los programas Java.
- La especificación de la JVM es un estándar. Cada sistema operativo tiene su propia implementación de la JVM.

API (Application Programming Interface)

La API JAVA es una colección de clases y otras componentes de software compiladas (archivos .class) que proveen una amplia gama de funcionalidades como componentes de GUIs, I/O, manipulación de colecciones, etc.

La API está agrupada en librerías de clases e interfaces Java relacionadas, llamadas paquetes. El programador puede combinar las componentes de la API JAVA con su código para crear una aplicación.

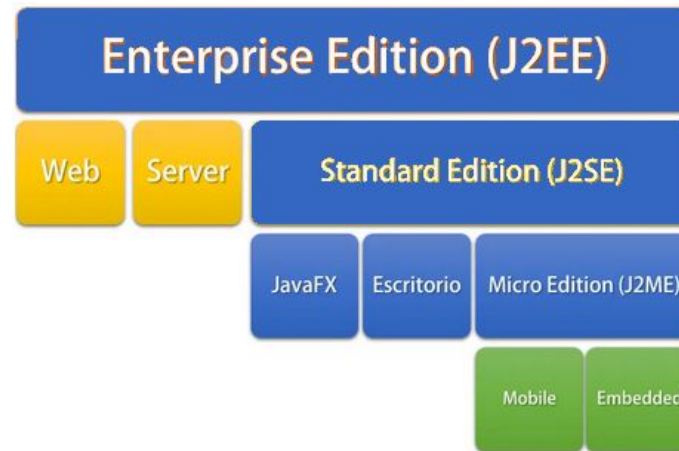


API (Application Programming Interface)

Algunos paquetes de la API JAVA son:

- *java.lang*: contiene clases esenciales como números, strings, objetos, seguridad y threads. Es el único paquete que se incluye automáticamente en todo programa Java.
- *java.io*: contiene las clases que manejan la Entrada/Salida, Serialización de objetos.
- *java.util*: contiene clases útiles que permiten manejar estructuras de datos o colecciones, fechas, hora, etc.
- *java.net*: contiene clases como URL, TCP, UDP, IP, etc. que permiten implementar aplicaciones distribuidas. Provee soporte para sockets.
- *java.awt*: contiene clases para el manejo de la GUI, pintar gráficos e imágenes.
- *java.applet*: contiene clases útiles para la creación y manipulación de applets y recursos para reproducción de audio.
- *java.sql*: contiene clases para el manejo de base de datos relaciones.

La Plataforma Java: ediciones

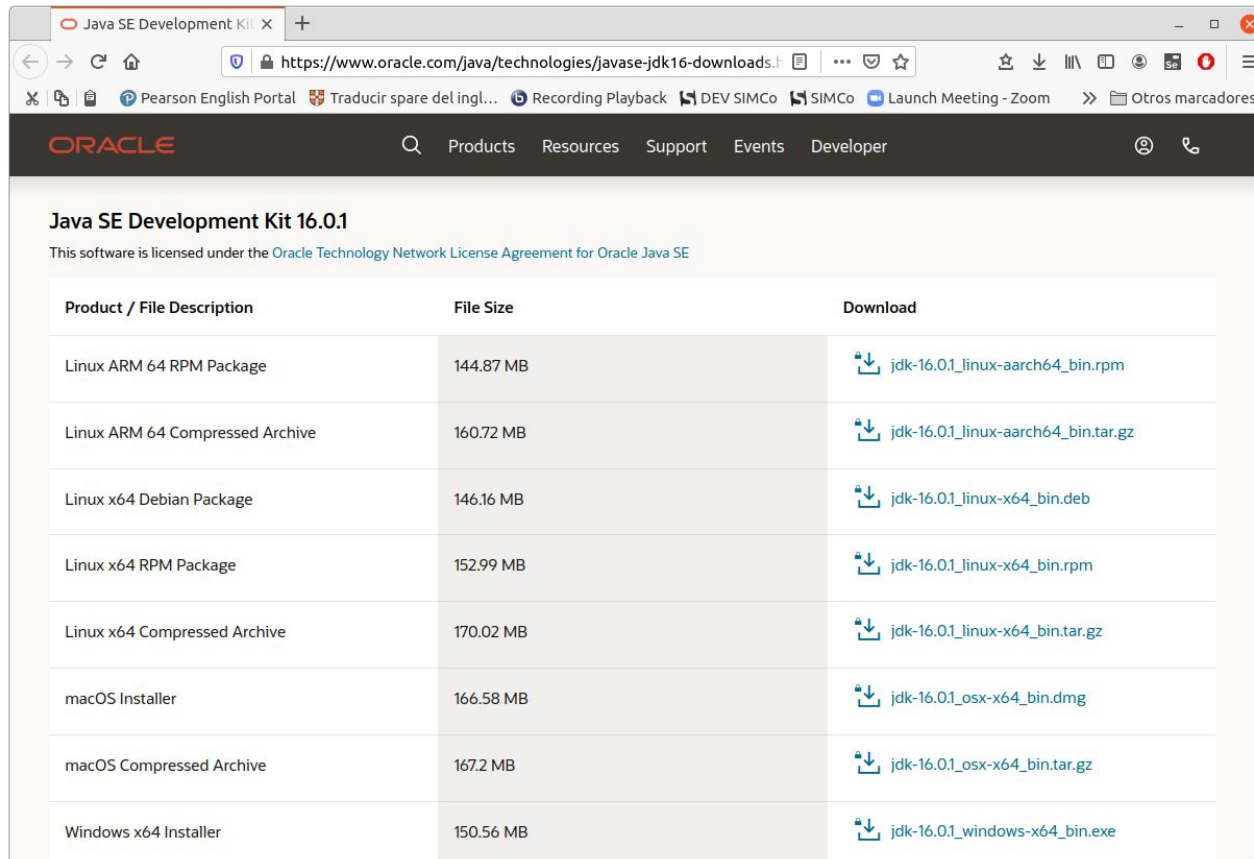


JEE (Java Enterprise Edition): está diseñada para programar y ejecutar aplicaciones empresariales, caracterizadas por ser multiusuario y distribuidas. El procesamiento de estas aplicaciones se realiza en un servidor. Usualmente son aplicaciones web.

JSE (Java Standard Edition): está diseñada para programar y ejecutar applets y aplicaciones de escritorio JAVA. Típicamente son programas que se ejecutan en una PC. Es el fundamento de las 2 restantes ediciones. Está compuesta por el JRE y el JDK.

JME (Java Micro Edition): está diseñada para programar y ejecutar aplicaciones para dispositivos con recursos de cómputo limitados, como los dispositivos móviles. Estos dispositivos cuentan con poca memoria RAM, pantallas muy chicas y muchas veces conexiones de red intermitentes.

Java SE Development Kit (JDK)



Java SE Development Kit 16.0.1		
This software is licensed under the Oracle Technology Network License Agreement for Oracle Java SE		
Product / File Description	File Size	Download
Linux ARM 64 RPM Package	144.87 MB	jdk-16.0.1_linux-aarch64_bin.rpm
Linux ARM 64 Compressed Archive	160.72 MB	jdk-16.0.1_linux-aarch64_bin.tar.gz
Linux x64 Debian Package	146.16 MB	jdk-16.0.1_linux-x64_bin.deb
Linux x64 RPM Package	152.99 MB	jdk-16.0.1_linux-x64_bin.rpm
Linux x64 Compressed Archive	170.02 MB	jdk-16.0.1_linux-x64_bin.tar.gz
macOS Installer	166.58 MB	jdk-16.0.1_osx-x64_bin.dmg
macOS Compressed Archive	167.2 MB	jdk-16.0.1_osx-x64_bin.tar.gz
Windows x64 Installer	150.56 MB	jdk-16.0.1_windows-x64_bin.exe

Desde esta URL se puede descargar la plataforma estándar de Java.

El Java SE Development Kit es la plataforma de desarrollo.

La especificación es única y existen implementaciones para cada plataforma.

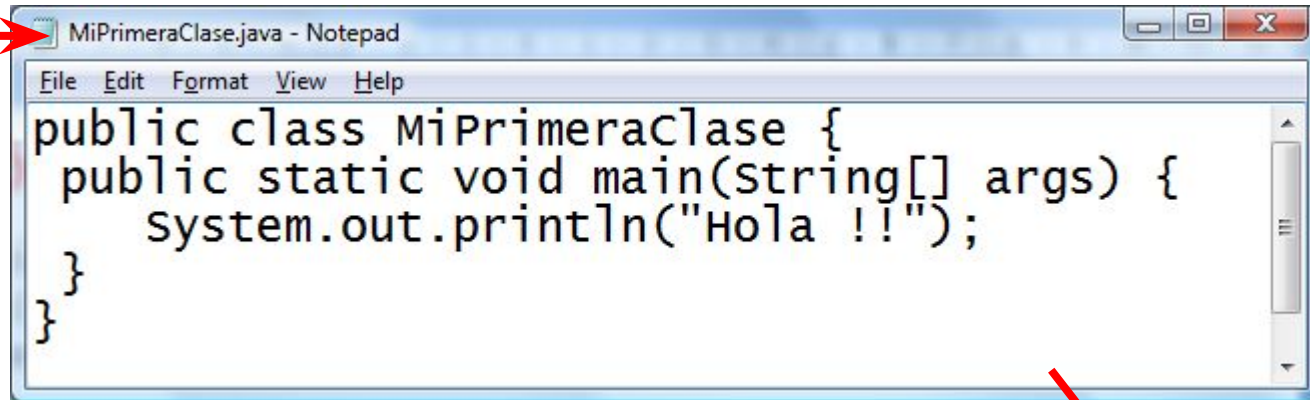
Oracle Java SE Product Releases: <https://www.oracle.com/java/technologies/java-se-support-roadmap.html>

Características del Lenguaje

- **Simple y Familiar:** JAVA tiene un núcleo consistente de conceptos fundamentales que facilita su aprendizaje. La sintaxis de JAVA es similar a la C++, aunque de menor complejidad.
- **Orientado a Objetos:** los programas JAVA están basados en objetos y en las interfaces para trabajar con esos objetos.
- **Robusto:** los programas JAVA son chequeados estrictamente mediante software antes de ejecutarse.
- **Seguro:** JAVA incorpora características de seguridad para garantizar que los programas que se ejecutan en una red no dañen los archivos de la computadora ni introduzcan virus.
- **Portable:** los programas JAVA pueden fácilmente ser transferidos de plataforma. Por ej. pueden pasar de ejecutarse en una PC con Windows a un servidor con Linux.
- **Alta performance:** los programas Java pueden ejecutarse eficientemente.
- **Multihilo:** soporta *threads* lo que permiten hacer múltiples tareas simultáneas logrando mejor tiempo de respuesta y comportamiento de los programas.
- **Dinámico:** los programas JAVA pueden adaptarse en ejecución a los cambios.

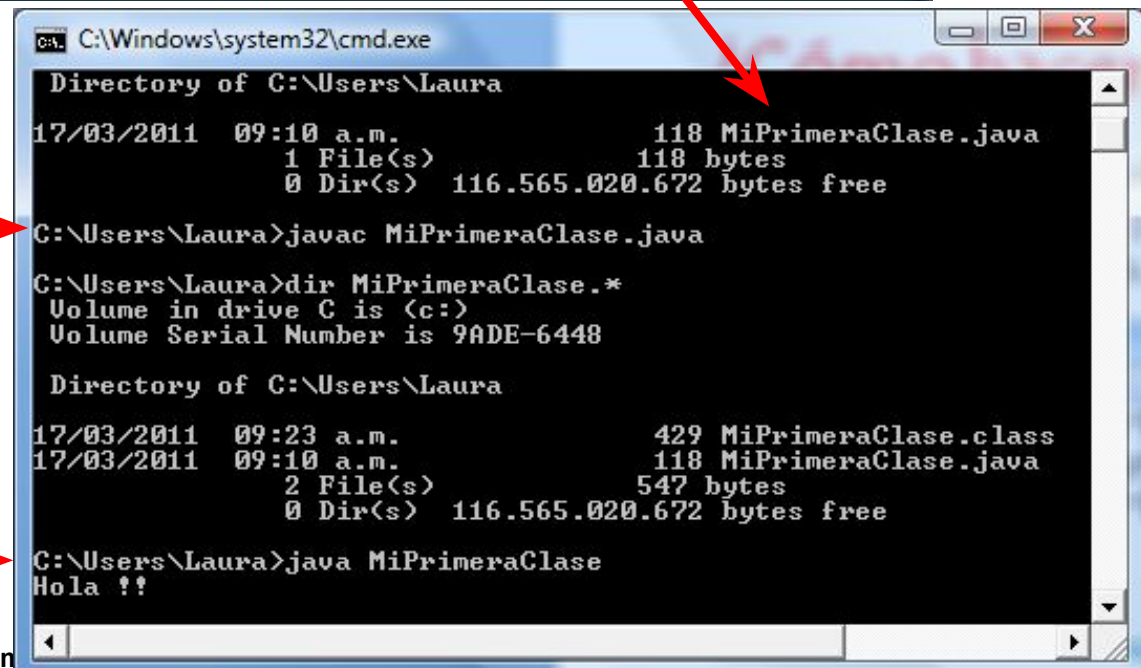
¿Cómo hacer y ejecutar el primer programa usando JAVA? (SIN IDE)

Un archivo fuente java, es un archivo de texto, que debe guardarse con el mismo nombre que la clase (y con extensión **.java**). Se deben respetar las mayúsculas. Para crear un programa en java, debemos crear una clase.



```
public class MiPrimeraClase {
    public static void main(String[] args) {
        System.out.println("Hola !!");
    }
}
```

¿Cómo se compila el
archivo fuente
java?



```
C:\Windows\system32\cmd.exe

Directory of C:\Users\Laura
17/03/2011  09:10 a.m.                118 MiPrimeraClase.java
               1 File(s)                118 bytes
               0 Dir(s)  116.565.020.672 bytes free

C:\Users\Laura>javac MiPrimeraClase.java

C:\Users\Laura>dir MiPrimeraClase.*
Volume in drive C is <c:>
Volume Serial Number is 9ADE-6448

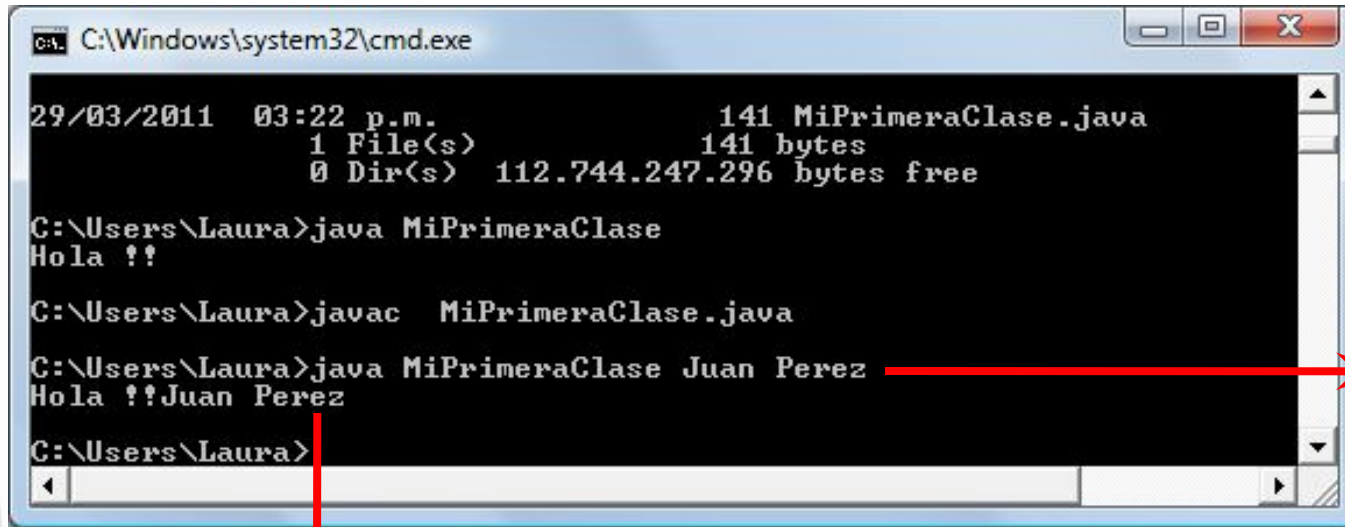
Directory of C:\Users\Laura
17/03/2011  09:23 a.m.                429 MiPrimeraClase.class
17/03/2011  09:10 a.m.                118 MiPrimeraClase.java
               2 File(s)                547 bytes
               0 Dir(s)  116.565.020.672 bytes free

C:\Users\Laura>java MiPrimeraClase
Hola !!
```

¿Cómo se ejecuta?

¿Cómo hacer y ejecutar el primer programa usando JAVA? (SIN IDE)

A un programa JAVA se le pueden pasar parámetros a través de la línea de comandos.



```
C:\Windows\system32\cmd.exe

29/03/2011  03:22 p.m.                141 MiPrimeraClase.java
          1 File(s)                  141 bytes
          0 Dir(s)  112.744.247.296 bytes free

C:\Users\Laura>java MiPrimeraClase
Hola !!

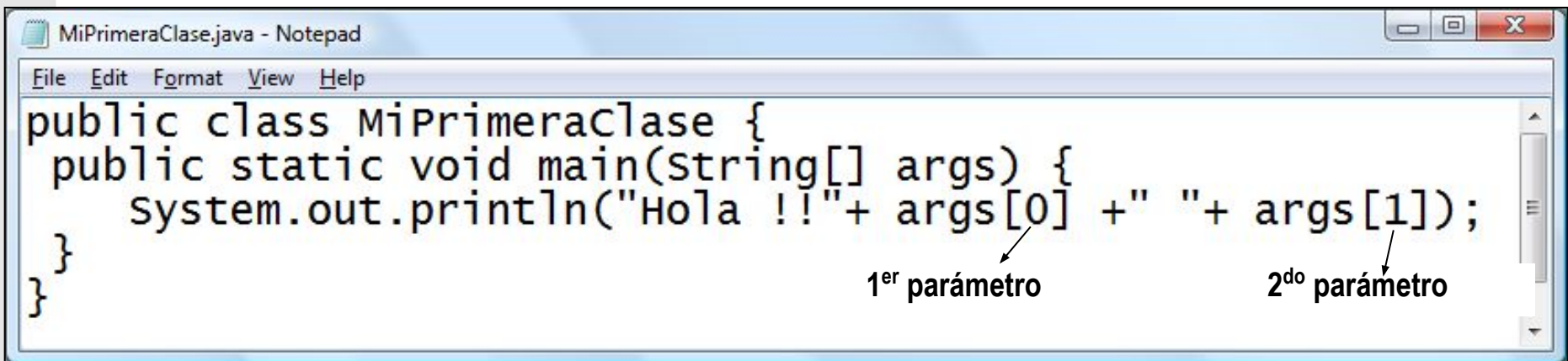
C:\Users\Laura>javac  MiPrimeraClase.java

C:\Users\Laura>java MiPrimeraClase Juan Perez
Hola !!Juan Perez

C:\Users\Laura>
```

Se pasan parámetros
Separados por blancos

¿Cómo hacemos para tomar desde el programa los parámetros de la línea de comandos?



```
MiPrimeraClase.java - Notepad
File Edit Format View Help

public class MiPrimeraClase {
    public static void main(String[] args) {
        system.out.println("Hola !!"+ args[0] +" "+ args[1]);
    }
}
```

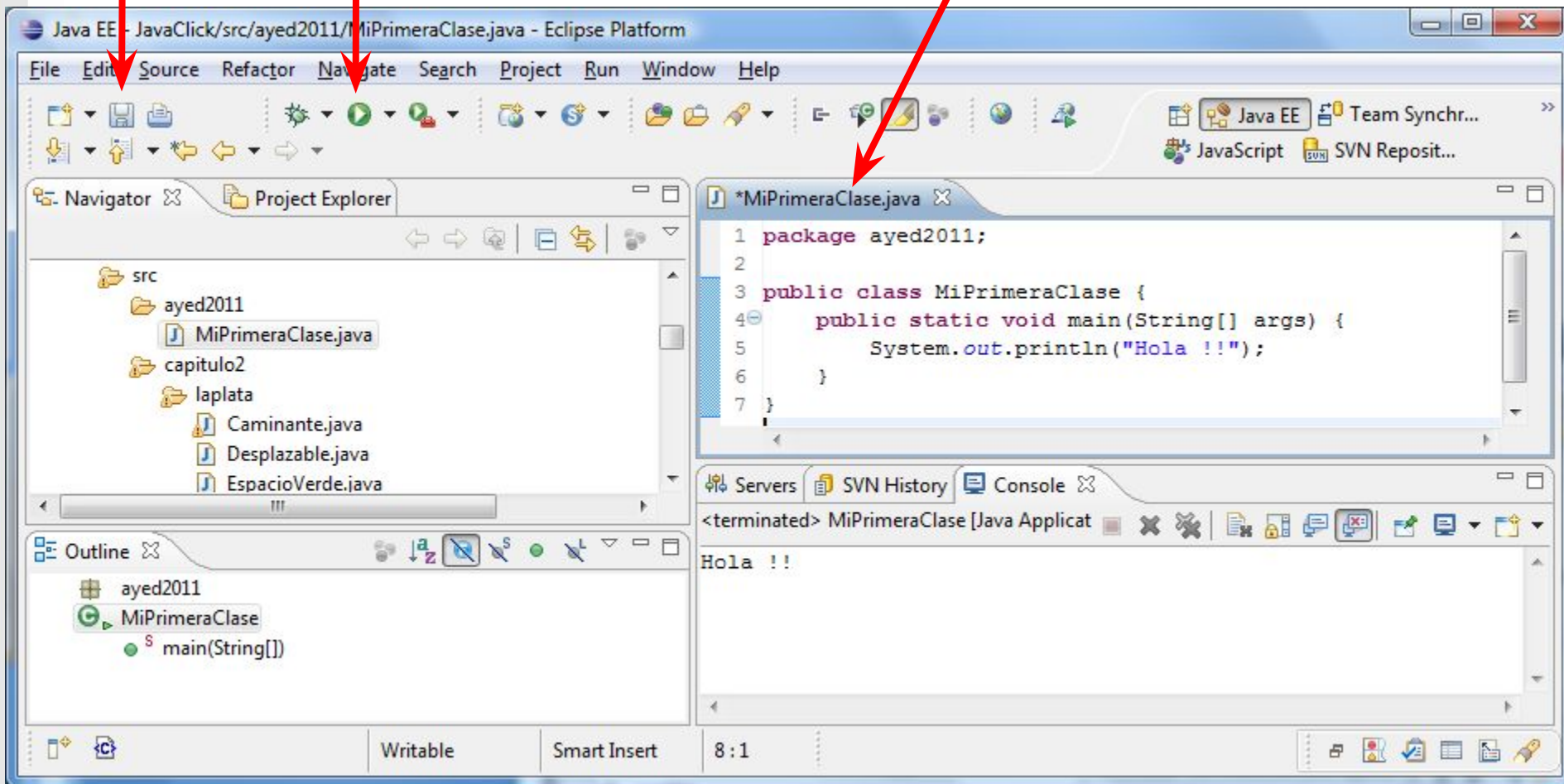
1^{er} parámetro 2^{do} parámetro

¿Cómo hacer y ejecutar el primer programa usando JAVA? (CON IDE)

Salva y compila

Ejecuta (run)

Nombre de la clase



¿Cómo pasamos parámetros a un programa desde Eclipse?

Se escriben los **parámetros** separados por blancos o uno debajo de otro

Run Configuration
(desde la barra de opciones o desde el menú contextual)

```
1 package ayed2011;
2
3 public class MiPrimeraClase {
4     public static void main(String[] args) {
5         System.out.println("Hola !!"+ args[0]+" "+ args[1]);
6     }
7 }
8
```

Run As

- 1 Run on Server
- 2 Java Application
- Run Configurations...

Run Configurations

Create, manage, and run configurations

Run a Java application

Name: Test

Program arguments:
Juan Perez

VM arguments:

Working directory:
☒ Default: \${workspace_loc:JavaClick}

Workspace... File System... Variables...

Apply Revert

Run Close

Filter matched 74 of 83 items