

Programación en Java

Tema 1. Introducción al lenguaje Java

Luis Rodríguez Baena

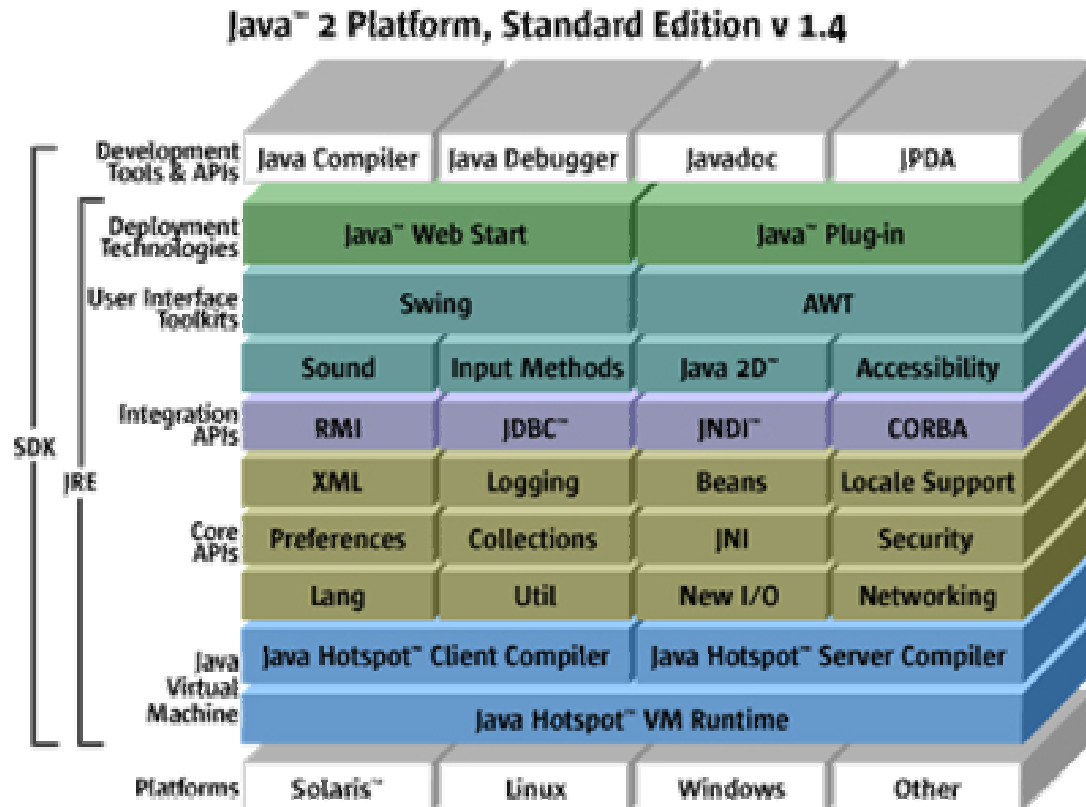
Universidad Pontificia de Salamanca (campus Madrid)

Facultad de Informática

Genealogía del lenguaje

- ❑ Sun Microsystems, 1991: Proyecto Oak.
- ❑ 1995: Primera versión JDK 1.0.
- ❑ 1997: JDK 1.1.
- ❑ 1998: JDK 1.2 (Java 2).
- ❑ 2000: JDK 1.3.
- ❑ Marzo 2002: JDK 1.4.
- ❑ Ediciones: J2SE, J2EE, J2ME.
 - Incluye JRE (Java Runtime Environment), utilidades de desarrollo (SDK), paquetes estándar.

Componentes de J2SE v1.4



Características generales (I)

❑ Simplicidad.

- Sintaxis parecida a C++, lo que implica una rápida curva de aprendizaje.

❑ Orientación a objetos.

- Lenguaje totalmente orientado a objetos.

❑ Portabilidad.

- Necesidad de un lenguaje multiplataforma para las aplicaciones en Internet a través de clientes *web* (navegador).
- El código **compilado** de Java se puede ejecutar sobre cualquier plataforma que implemente una Máquina Virtual Java (JVM).

Características generales (II)

❑ Concurrencia (multihilo, *multi-thread*).

- Permite la ejecución de varios hilos de proceso (*thread*) que se comunican entre si para permitir su colaboración.
- La concurrencia permite por ejemplo:
 - ✓ Evitar las esperas de la interfaz de usuario.
 - ✓ Recibir información de forma asíncrona (descarga de archivos).
 - ✓ Aplicaciones multiusuario.
 - ✓ Aplicaciones multiproceso si el *hardware* y los sistemas operativos lo permiten.

❑ Distribuido y dinámico.

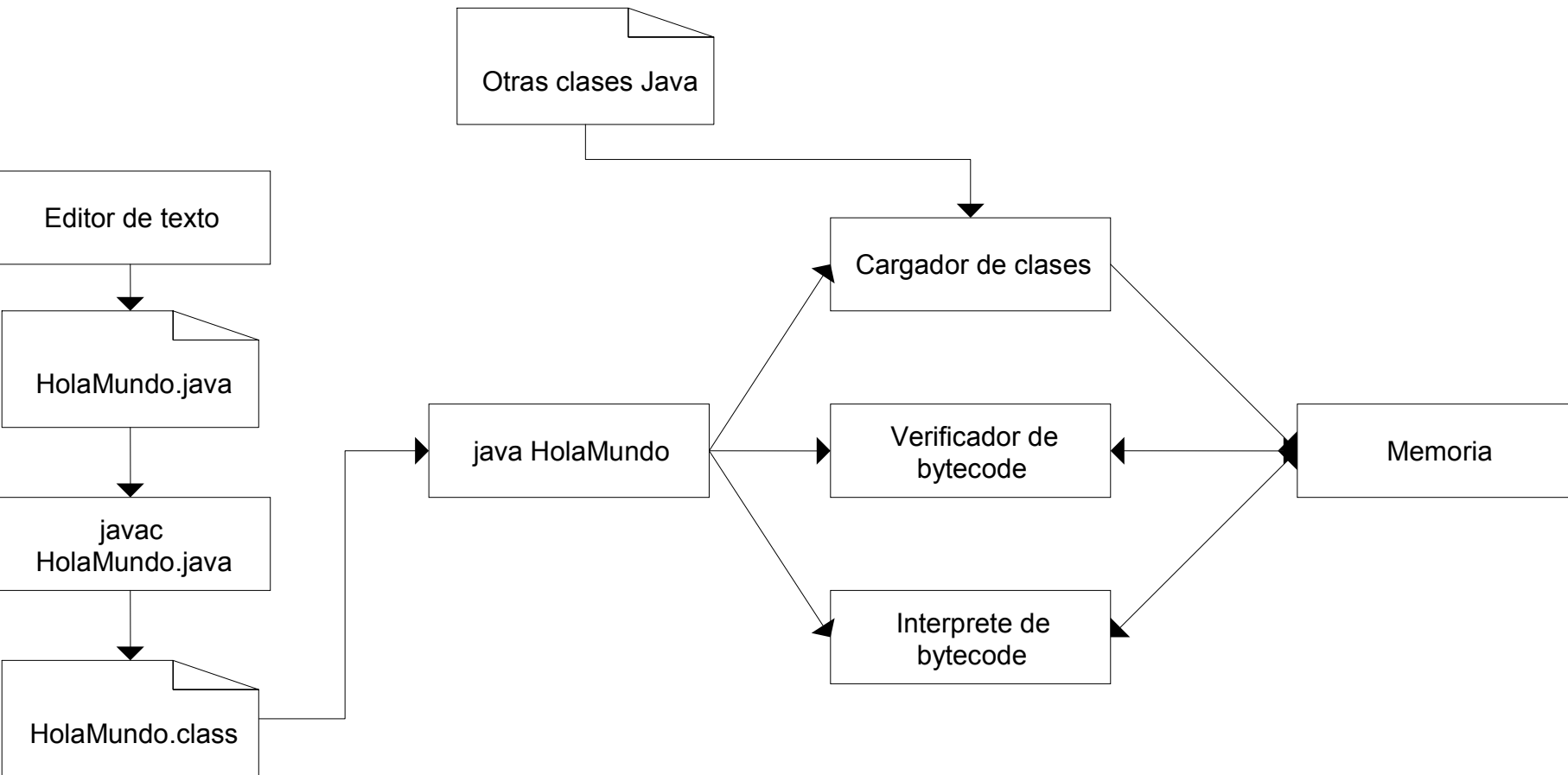
- Carga dinámica de bibliotecas de clases.
- Aplicaciones con objetos distribuidos RMI (*Remote Method Invocation*), CORBA, *Enterprise JavaBeans* (EJB).

Características generales (III)

❑ Seguridad

- Tres componentes del modelo de seguridad:
 - ✓ Verificador de bytecode.
 - ✓ Cargador de clases.
 - ✓ Security Manager
- Seguridad en los applets
 - ✓ Applets firmados.
- APIs de criptografía, certificados digitales, sistema de claves, generador de claves.

El modelo de compilación (I)



El modelo de compilación (II)

□ Edición del archivo fuente con un editor de texto.

```
import java.util.*;
class HolaMundo {
    public static void main(String args[]) {
        System.out.println("¡Hola mundo!");
        Date d = new Date();
        System.out.println("Hoy es " + d);
    }
}
```


El modelo de compilación (III)

□ Compilación

- `javac HolaMundo.java`
- `javac.exe` genera bytecode independiente de la plataforma y ejecutable en una JVM.
- Se genera un archivo `.class` por cada clase presente en la unidad de compilación.
- Posibles ajustes necesarios:
 - ✓ Variable de entorno `PATH`
 - ✓ Variable de entorno `CLASSPATH`

Modelo de compilación (IV)

❑ Ejecución

- Ejecución del archivo `.class` mediante el interprete `java.exe` (`java HolaMundo`).
 1. Cargador de clases.
 2. Verificador de bytecode.
 3. Unidad de ejecución
- Interpretación = merma de eficiencia.
 - ✓ Compilador JIT (*Just In Time*).
 - ✗ Convierte el bytecode a código nativo que ejecuta la JVM.
 - ✓ Compiladores en código nativo.
 - ✗ Pérdida de compatibilidad.

Modelo de compilación (V)

❑ Entornos de desarrollo Java.

- Visual Age de IBM.
- JBuilder de Borland.
- Visual Café de Symatec.
- Visual J++ de Microsoft.

❑ Algunos errores de compilación frecuentes

- javac Bad command or file name.
- Holamundo.java:12: ';' expected

```
System.out.println(";Hola mundo!")
```

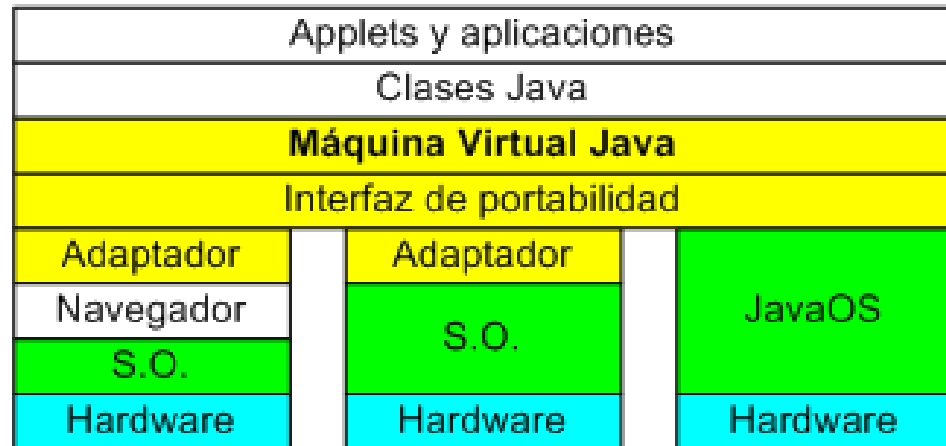
^
- Holamundo.java:10: cannot resolve symbol
symbol : method printl (java.lang.String)
location: class java.io.PrintStream

```
System.out.printl(";Hola mundo!");
```

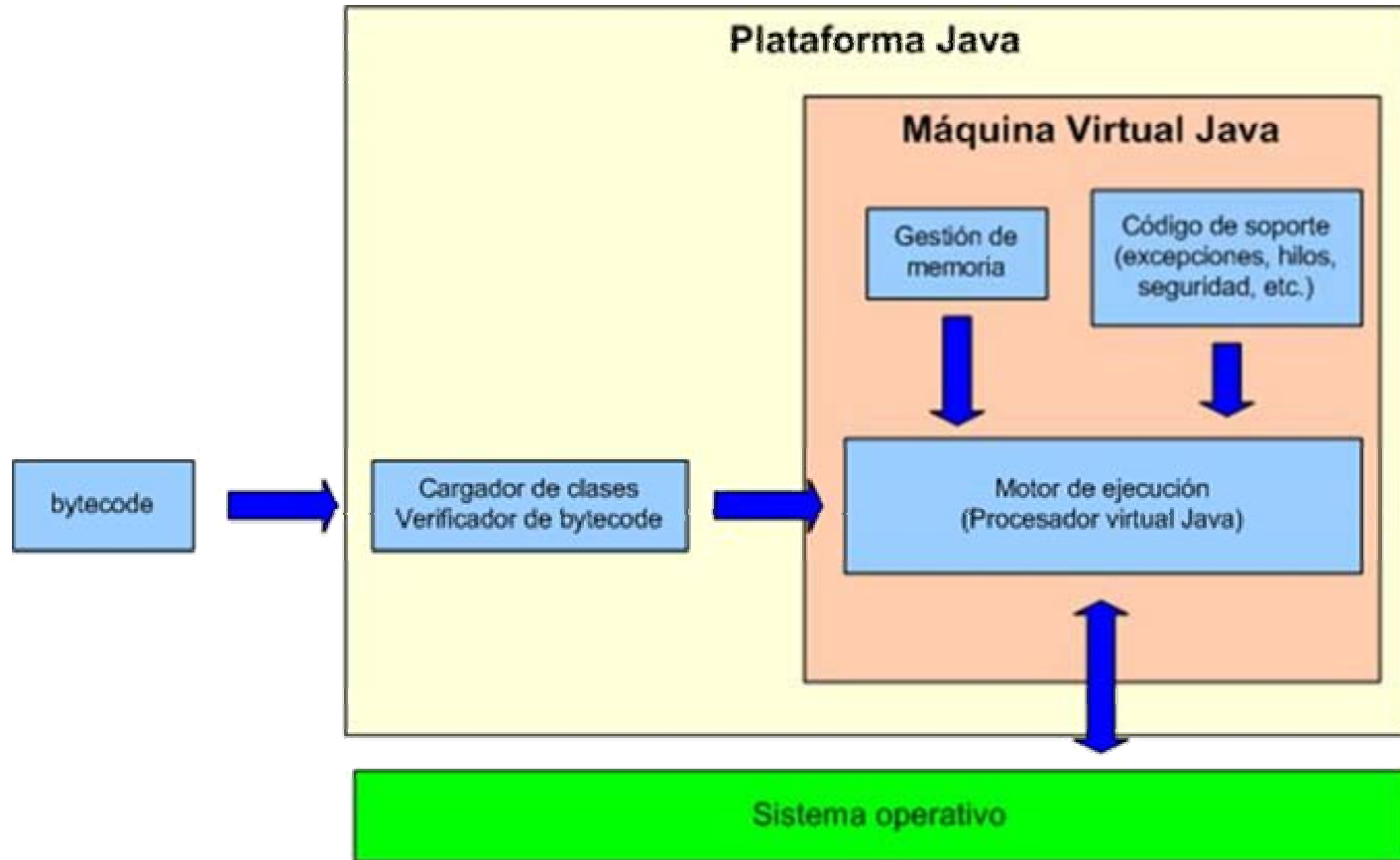
^
- Exception in thread "main" java.lang.NoClassDefFoundError:
HolaMundo

Arquitectura de la Máquina Virtual Java (I)

- ❑ JVM: capa interpuesta entre el sistema operativo y las clases Java.
- ❑ Interpreta el bytecode a las órdenes nativas de un entorno (*hardware* o *software*) determinado.
- ❑ Dos capas:
 - Dependiente de la plataforma: **adaptador**.
 - Independiente de la plataforma: **interfaz de portabilidad**.
 - Sólo es necesario escribir un nuevo adaptador para cada plataforma.



Arquitectura de la Máquina Virtual Java (II)



Utilidades principales del SDK

❑ javac.

- Compilador de Java. Toma una unidad de compilación (.java) y convierte las clases contenidas en archivos .class.

❑ java.

- Interprete Java. Verifica, carga y traduce el bytecode de un archivo .class.

❑ appletviewer.

- Interprete que ejecuta los applets contenidos en un archivo HTML.

❑ javadoc.

- Crea documentación en formato HTML a partir del código y los comentarios de un programa Java.