5. Errores en nuestros programas

A todos nos gustaría nacer con las cosas ya aprendidas, y saber hacerlo todo bien a la primera. Sin embargo, lo cierto es que siempre hay que dedicar mucho tiempo y duros esfuerzos para llegar a ser buenos en lo que nos propongamos. En este apartado vamos a dar las primeras pautas de cómo funciona, en Java, la compilación del código fuente (guardado en los archivos .java), que nos permitirá convertirlo en un código intermedio compilado llamado bytecode (que tendrá el mismo nombre del archivo del código fuente, pero cambiará la extensión a .class). Y finalmente, cuando queramos ejecutar este código en una máquina virtual Java, esta interpretará el bytecode libre de errores y lo traducirá al lenguaje máquina de ese dispositivo electrónico en el que se esté ejecutando.

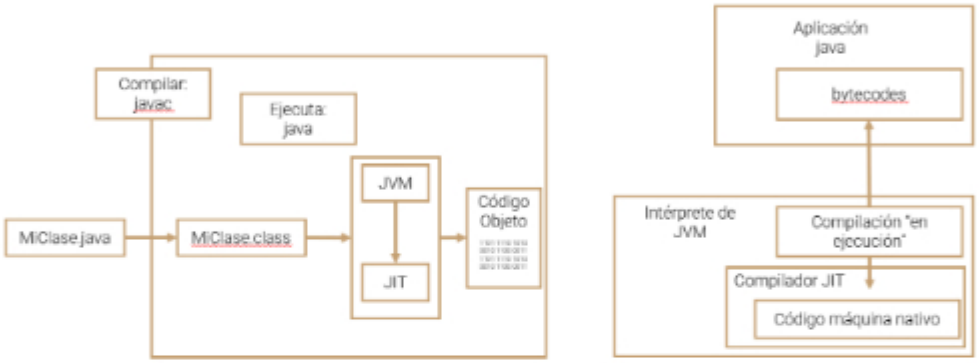

Figura 1.13. Pasos de compilación e interpretación del código Java

Por tanto, durante la etapa de desarrollo de nuestro programa pueden darse dos tipos de errores: los relativos al tiempo de compilación y los que afectan al tiempo de ejecución.

5.1. Errores en el tiempo de compilación

Cada vez que ejecutemos el compilador en Java sobre nuestro código fuente va a realizar las siguientes comprobaciones, y cuando encuentre un error se detendrá para mostrárnoslo:

- Análisis a nivel **léxico-gráfico**: leerá carácter a carácter lo que hemos escrito, pasando a agruparlo en palabras que Java denomina tokens. Estos han de estar definidos por el lenguaje o por nosotros. Si encuentra alguna palabra que puede ser una frase sin espacios no definida en ningún sitio, nos mostrará un error. Por ejemplo, si al escribir la palabra reservada «class» nos despistamos y dejamos un espacio en blanco en medio, Java creerá que son dos tokens diferentes que no están definidos y, por tanto, dará un error y lo marcará en el depurador que utilicemos.

```
1 public cla ss HolaMundo
2 {
3     public static void main(String[] args) {
4         System.out.println("Hello World");
5     }
6 }
```

```
Compilation failed due to following error(s).
Main.java:1: error: class, interface, or enum expected
public cla ss HolaMundo
           ^
```

Figura 1.24. Error léxico: el token de la clase tiene un error.

- Análisis a nivel **sintáctico**: vamos un paso más allá, ahora Java revisará la combinación de esos tokens para dar significado a una estructura o a un método. El lenguaje sabe que, al crear

una clase, se espera un identificador creado por el programador, pero sino lo encuentra y aparece una { que abre el bloque de la clase, habrá detectado ese error. Inmediatamente, parará de compilar y nos indicará dónde lo ha encontrado.

```
1 public class
2 {
3     public static void main(String[] args) {
4         System.out.println("Hello World");
5     }
6 }
```

```
Compilation failed due to following error(s).
Main.java:1: error:  expected
public class
             ^
1 error
```

Figura 1.35. Error sintáctico: falta el nombre de la clase

- Análisis a nivel **semántico**: Y por último se fijará en aspectos del significado, sentido o interpretación de los tokens al combinarlos entre sí. Por ejemplo, nosotros en castellano podemos decir: «El gato acaba de ladrar». Esta frase pasaría los dos análisis previos, pero este último no lo pasaría, ya que todos sabemos que los que ladran son los perros, no los gatos. Cuando empezamos a utilizar condiciones en nuestro código, podrán aparecer estos errores. O si queremos imprimir con println() una variable que no se ha inicializado y, por tanto, no tiene aún ningún valor, en tal caso también nos dará este tipo de error. Lo iremos viendo en las próximas unidades.

5.2. Errores en el tiempo de ejecución

Los errores en el tiempo de ejecución pueden deberse a situaciones inesperadas (como una división por cero en alguna operación o a alguna otra que dé un resultado indeterminado y no tenga solución) o bien a errores que quiere gestionar el propio programador.

Por ejemplo, si vamos a programar un sistema de votación electrónica, querremos que solo los mayores de edad puedan votar en las elecciones utilizando un certificado digital desde su teléfono inteligente. Si intenta votar alguien que aún no haya cumplido los 18 años o que no tenga la edad mínima fijada en los requisitos de nuestro cliente, no podrá votar.

Nosotros, como programadores, podremos provocar errores en el tiempo de ejecución para controlar estas situaciones erróneas, que en Java se denominan excepciones. En futuras unidades veremos cómo provocarlos, gestionarlos y arreglarlos en tiempo de ejecución. Esta característica es fundamental en cualquier lenguaje de programación actual.

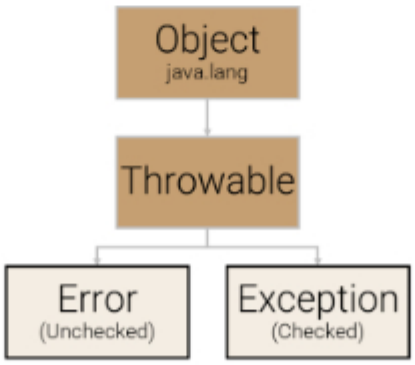

Figura 1.16. Tipos de errores en tiempo de ejecución