---

# Novedades: Introducción


Java ha ido evolucionando y añadiendo características nuevas con cada versión, algunas de ellas bastante curiosas y modernas. Aunque no son imprescindibles para empezar a programar, es interesante conocerlas. Quizás en el futuro te encuentres con alguna y al menos deberías saber que existen.

Por ahora, tómalo más como anécdotas tecnológicas que como herramientas esenciales. Eso sí, como verás, algunas pueden ser útiles, pero otras podrían complicar un poco las cosas mientras aprendes. ¡Vamos a descubrir algunas!

# Declaración de variables con `var`

En las últimas versiones de Java (a partir de **Java 10**), se introdujo la palabra clave `var` para declarar variables de una forma más sencilla y concisa. Aquí vemos cómo funciona.

## ¿Qué hace `var`?

La palabra clave `var` permite que Java adivine automáticamente el **tipo de la variable** en el momento en que la declaras. En lugar de escribir el tipo de la variable explícitamente (como `int`, `String`, etc.), puedes usar `var` y dejar que Java lo deduzca a partir del valor que le asignas.

## Ejemplo con `var`

```
var numero = 10;        // Java sabe que 'numero' es un int
var texto = "Hola";     // Java sabe que 'texto' es un String
var decimal = 3.14;     // Java sabe que 'decimal' es un double
```

> **Note**
>
> 😳JavaScript... ¿eres tú?

En estos ejemplos:

- Java **deduce** que `numero` es de tipo `int` porque le asignamos un número entero.
- Java **deduce** que `texto` es un `String` porque le asignamos un texto entre comillas.
- Java **deduce** que `decimal` es un `double` porque le asignamos un número con decimales.

## ¿Qué cambia con `var`?

- **Menos código repetido**: Antes, tenías que escribir el tipo dos veces (una al declarar y otra al asignar). Con `var`, solo te enfocas en el valor.

  Ejemplo tradicional:

  ```
  int numero = 10;
  String texto = "Hola";
  ```

  Con `var`:

  ```
  var numero = 10;
  var texto = "Hola";
  ```

## Lo que NO hace `var`

- **No es una variable dinámica**: A pesar de que `var` parece hacer que Java sea más flexible, el **tipo de la variable no cambia**. Una vez que Java decide el tipo de la variable, ese tipo queda fijo.

  Por ejemplo, si `var numero = 10;`, `numero` será un `int` y no podrás asignarle, más adelante, un valor que no sea de tipo `int`.

## Reglas importantes

- **Siempre debe estar inicializada**: Java necesita saber el tipo en el momento en que creas la variable, así que no puedes declarar una variable con `var` sin asignarle un valor.

```
var x;  // ERROR: Java no sabe qué tipo es 'x'
```

- **El código sigue siendo fuertemente tipado**: Aunque no declares el tipo explícitamente, el compilador lo hace por ti. Por ejemplo, si asignas un número a una variable con `var`, esta variable siempre será de tipo numérico.

## Cuándo usar `var`

- Cuando el tipo de la variable es obvio y quieres evitar escribirlo de manera redundante.
- Para hacer el código más limpio y fácil de leer.

## Cuándo evitar `var`

- Si el tipo de la variable no es claro solo con mirarla. Por ejemplo, en casos donde podría haber confusión sobre qué tipo es la variable (como en métodos más complejos), es mejor declarar el tipo explícitamente.
- **Cuando estás aprendiendo.** De la forma tradicional, Java nos obliga a tener en mente el tipado de datos siempre.

> **Tip**
>
> Aunque `var` simplifica la escritura de código, es recomendable evitarlo cuando estás empezando. Usar el tipo explícito (como `int`, `String`, etc.) te ayuda a comprender mejor el tipado de las variables y a evitar confusiones sobre qué tipo de dato estás manejando en cada momento.

---

# 👷 En construcción 🚧

Esta sección está en constante evolución. Aquí iremos creando nuevas secciones cuando aparezcan o descubramos novedades importantes sobre las últimas versiones de java.