Java 12-17

Profesor:
Máster Carlos Carranza Blanco
Programación III
Ingeniería del Software

Nuevas Características Java 12 (2019.3)

- *Las características destacadas de Java 12 son la incorporación de forma experimental las expresiones switch y mejoras en el recolector de basura para mayor rendimiento.*

- *Formato de número compacto:*

```
NumberFormat number = NumberFormat.getCompactNumberInstance(Locale.US, NumberFormat.Style.SHORT);
System.out.println(number.format(1000));
```

Resultado: 1K

- *En el siguiente enlace pueden ver algunas de las novedades más relevantes:*
- *https://picodotdev.github.io/blog-bitix/2019/03/novedades-de-java-12/*

Nuevas Características Java 13 (2019.9)

- *Java 13 incorpora algunas nuevas características interesantes que mejoran un facilitan la lectura del código, entre las más destacadas están los bloques de texto y las expresiones switch mejoradas.*
- *Casi coincidiendo con la publicación de Java 13 se publicado Jakarta EE 8 que únicamente tiene como novedad que su propiedad ha pasado a estar baja la fundación Eclipse, es totalmente compatible con Java EE 8 y sus últimas mejoras, las novedades vendrán en versiones posteriores de Jakarta EE en las que se dará importancia a la tendencia de las aplicaciones para su funcionamiento en entornos orientados a la nube.*
- *Casi al mismo tiempo se ha publicado JavaFX 13 ya fuera del JDK en donde puede seguir su propio ciclo de publicaciones independiente del JDK.*

- *En el siguiente enlace pueden ver algunas de las novedades más relevantes:*
- *https://picodotdev.github.io/blog-bitix/2019/09/novedades-de-java-13/*

Bloques de texto

Para definir una cadena de caracteres que tuviese varias lineas en Java había que emplear concatenación de cadenas, si esa cadena contiene el caracter comilla doble " de inicio de cadena había que escaparlo, si esa cadena contenía saltos de línea había que emplear el caracter de escape de salto de línea \n. El resultado es una cadena con problemas de legibilidad por los caracteres de escape que incluye en el código fuente del lenguaje. Esto podría se al definir una cadena de texto que tuviese elementos HTML, JSON, sentencias SQL o expresiones regulares.

```
1 String html = "<html>\n" +
2     "    <body>\n" +
3     "        <p class=\"text\">Hello, Escapes</p>\n" +
4     "    </body>\n" +
5     "</html>\n";
```

TextBlock-1.java

Con los bloques de texto se emplean una triple comilla doble """ para la apertura y cierre de la cadena.

```
1 String html = """
2     <html>
3         <body>
4             <p class="text">Hello, Text Blocks</p>
5         </body>
6     </html>""";
```

TextBlock-2.java

Como ayuda a las cadenas de texto en la clase String se han añadido varios métodos para eliminar la indentación (*String::stripIndent*), traducir los caracteres secuencia de escape (*String::translateEscapes*) y formatear una cadena usando un método de instancia (*String::formatted*).

Expresiones switch mejoradas

En las novedades de Java 12 se añadió la posibilidad de los *switch* fueran expresiones que retornan un valor en vez de sentencias y se evita el uso de la palabra reservada *break*.

```
1 String numericString = switch(integer) {
2   case 0 -> "zero";
3   case 1, 3, 5, 7, 9 -> "odd";
4   case 2, 4, 6, 8, 10 -> "even";
5   default -> "N/A";
6};
```

Switch-1.java

En Java 13 en vez únicamente el valor a retornar se permite crear bloques de sentencias para cada rama *case* y retornar el valor con la palabra reservada *yield*. En los bloques de sentencias puede haber algún cálculo más complejo que directamente retornar el valor deseado.

```
1 String numericString = switch(integer) {
2   case 0 -> {
3       String value = calculateZero();
4       yield value;
5   } ;
6   case 1, 3, 5, 7, 9 -> {
7       String value = calculateOdd();
8       yield value;
9   };
10  case 2, 4, 6, 8, 10 -> {
11      String value = calculateEven();
12      yield value;
13  };
14  default -> {
15      String value = calculateDefault();
16      yield value;
17  };
18};
```

Switch-2.java

Nuevas Características Java 14 (2020.3)

- *Entre las novedades más destacadas que incorpora Java 14 están los records(Preview), la incorporación definitiva de las expresiones switch o el pattern matching para el operador instanceof. Otra de las novedades más destacadas es una traza de NullPointerException más útil, también destaca la posibilidad de utilizar el recolector de basura ZGC en Windows y macOS.*
- *En el siguiente enlace pueden ver algunas de las novedades más relevantes:*
- *https://picodotdev.github.io/blog-bitix/2020/03/novedades-de-java-14/*

Excepciones NullPointerException más útiles

- *Hay casos en los que la trazas de NullPointerException no es lo suficientemente precisa para determinar la causa de la excepción sin usar el debugger. En los siguientes ejemplos con elementos encadenados no es posible determinar cuál es la variable que ha originado la excepción por tener valor nulo.* a.b.c.i = 99;

- *A partir de Java 14 las excepciones NullPointerException son más útiles e indican de forma precisa cual es el miembro de la línea de código que ha producido la excepción.*

Nuevas Características Java 15 (2020.9)

- *Algunos cambios no tienen gran impacto en el lenguaje ni la plataforma al ser más cambios internos que reimplementan y modernizan código existente, otras son versiones preliminares y segundas versiones preliminares no definitivas, algunas características en versiones anteriores se marcan como públicas dejando de ser preliminares, otras que se marcan como obsoletas desaconsejándose su uso y otras ya marcadas como obsoletas anteriormente son eliminadas.*
- *En el siguiente enlace pueden ver algunas de las novedades más relevantes:*
- *https://picodotdev.github.io/blog-bitix/2020/09/novedades-de-java-15/*

Nuevas Características Java 16 (2021.3)

- *No hay cambios importantes en el lenguaje, el más destacado es la inclusión de pattern matching para el operador instanceof, la incorporación de forma final de los Records, soporte para las plataformas Windows/AArch64, distribuciones como Alpine basadas en musl y soporte para la comunicación mediante canales Unix-Domain Socket.*
- *En el siguiente enlace pueden ver algunas de las novedades más relevantes:*
- *https://picodotdev.github.io/blog-bitix/2021/03/novedades-de-java-16/*

Packaging Tool

- *Se proporciona una herramienta para empaquetar aplicaciones Java que genera instaladores nativos para las diferentes plataformas. La herramienta jpackage fue incorporada en Java 14 y ahora pasa a considerarse con la categoría de listo para producción, su API está en el jdk.jpackage.*

- *Soporta los formatos de empaquetado nativos de la plataforma para tener una experiencia de instalación natural. Estos formatos incluyen archivos msi y exe en windows, pkg y dmg en macOS, deb y rpm en Linux. Permite especificar en tiempo de empaquetado parámetros a usar en tiempo de ejecución. Se puede invocar de forma directa desde la línea de comandos y de forma programática mediante su API.*

Pattern Matching para instanceof

- *Se mejora el soporte del lenguaje para soportar pattern matching en el operador instanceof. Esto evita la necesidad de realizar cast de forma explícita simplificando el código, más legible y seguro.*

```
1  // Antes
 2if (obj instanceof String) {
 3    String s = (String) obj;
 4    System.out.println(s);
 5}
 6
 7// Ahora, con pattern matching
 8if (obj instanceof String s) {
 9    System.out.println(s);
10}
11if (obj instanceof String s && s.contains("jdk")) {
12    System.out.println(s);
13}
```

Records

- *Las clases Record son un nuevo tipo de clases en el lenguaje Java, ayudan a modelar agregados de datos con menos ceremonia que las clases normales. Son clases que actúan como contenedores para datos inmutables, pueden ser considerados como tuplas. La declaración de un record mayormente consiste en la declaración de su estado. No es su objetivo resolver los problemas de las clases mutables que usan las convenciones de nombres de los JavaBeans.*
- *Se publicaron como primera vista previa en Java 14 y una segunda versión en Java 15.*
- **public** record **Point**(**int** x, **int** y) {}

***ver link**

Nuevas Características Java 17 (2021.9)

- *La versión Java 17 sucede a Java 11 como versión LTS, por ello es una versión más importante que las no LTS anteriores. Incorpora todas las mejoras incluidas en todas las no LTS previas más otras adicionales dede Java 16 publicada seis meses antes. Como versión LTS ofrece un soporte de correcciones de errores, fallos y alertas de seguridad por un periodo de cinco años hasta septiembre de 2026 más un periodo adicional de tres años hasta 2029. La versión 6 del framework de Spring y Spring Boot 3 se tendrán como requisito mínimo Java 17 y Jakarta 9.*
- *En el siguiente enlace pueden ver algunas de las novedades más relevantes:*
- *https://picodotdev.github.io/blog-bitix/2021/09/novedades-y-nuevas-caracteristicas-de-java-17/*

Encapsulación fuerte de las clases internas del JDK

- *Se encapsula de forma fuerte impidiendo su uso de todos los elementos internos del JDK de los paquetes java.*, sun.*, com.sun.*, jdk.* y org.* , exceptuando ciertas APIs críticas como sun.misc.Unsafe.*
- *La encapsulación fuerte permite mejorar la seguridad y el mantenimiento del JDK que era uno de los objetivos primarios del proyecto Jigsaw con la incorporación de los módulos.*

Clases sealed

- *Las clases sealed fueron propuestas en Java 15 en modo vista previa, en Java 16 fueron propuestas de nuevo con algunos cambios. Las clases sealed son incorporadas de forma final sin cambios respecto a Java 16. Las clases sealed permite limitar que que clases tienen permitido heredar de una clase definida como sealed.*

```
1 public abstract sealed class Shape permits Circle, Rectangle, Square { ... }
2 public class Circle extends Shape { ... }
3 public class Rectangle extends Shape { ... }
4 public class Square extends Shape { ... }
5
```

Resumen

- *El ciclo de publicación de una nueva versión de Java cada seis meses está siendo un éxito en la evolución del lenguaje, algunas características no tienen un gran impacto en la plataforma o el lenguaje pero otras sí suponen un gran mejora como las lambdas de Java 8, los módulos de Java 9, inferencia de tipos de Java 10, Java 11 como primera versión LTS con soporte a largo plazo, expresiones switch de Java 12 en vista previa, bloques de texto de Java 13 en vista previa, excepciones NullPointerException más útiles y Records de Java 14, Sealed Classes de Java 15 y encapsulación más fuerte de clases internas del JDK en Java 16 por mencionar simplemente una característica destacada de cada una de ellas.*
- *Java 17 al ser una versión LTS es más importante que las versiones no LTS anteriores. Las versiones LTS son más atractivas para ciertas organizaciones por sus periodos de soporte extendidos.*

Enlaces de interés

- *https://reflectoring.io/java-release-notes/*
- *https://docs.oracle.com/en/java/javase/20/language/java-language-changes.html#GUID-B06D7006-D9F4-42F8-AD21-BF861747EDCF*