Java 10

Denys Adrian Dutan Sanchez
Universidad Politecnica Salesiana
Azuay, Cuenca, Paute.
ddutans1@est.ups.edu.ec

Palabras claves—recolector de basura, API (aplicaction programming interfqce), doclet, y API DOM.

I. Objetivo

El objetivo de este informe es describir todo detalle que la aplicación de Java 10 haya incorporado en su nueva actualización. Se va a describir mas los nuevos métodos que se hayan instalado en la actualización, de la misma manera también se discutirá los métodos que se hayan eliminado en esta actualización. Se debe tener en cuenta que la aplicación de java al actualizar su sistema también analiza que métodos son menos útiles y realizan un deprécate del método. Se fijarán todos estos puntos a continuación.

II. Estado de arte

A. *Lanzamiento de Java 10 (Informe de actualizacion)*

Oracle lanzo la nueva actualización de java en 20 de marzo del año 2018. Algunas de sus mejores actualizaciones en esta versión de Java 10 incluye: el mejoramiento de la recolección y compilación de basura, también el mejoramiento de los tipos de variables locales de java.

B. *Recolector de basura*

La recolección de basura en Java es el proceso mediante el cual los programas de Java realizan la administración automática de memoria. Todos los programas de Java se compilan en un código de bytes que se ejecutan en una maquina virtual de Java. Cuando estos programas se ejecutan en Java los objetos se crean en un montón y estos son una parte de la memoria dedicada al programa. Eventualmente algunos de estos objetos ya no serán necesarios y aquí es donde el recolector de basura encuentra estos objetos y los elimina para liberar memoria.

C. *API DOM*

Un API DOM es una interfaz de programación de aplicaciones para HTML valido y documentos XML bien formados.

III. Actualizacion

A. *Abreviaciones y acronimos*

JDK: Java Development Kit

API: Application Programming Interface

DOM: Document Object Model

B. *Actualizaciones*

- Java ha establecido una Interferencia de tipos variable local, para mejorar el lenguaje de java para extender el tipo de interferencia de declaraciones de variables locales y inicializadores.
- Recolección de basura en paralelo para el recolector de basura G1, para mejorar las latencias del peor de los casos
- Tres nuevas opciones de JVM, para brindar a los usuarios de contenedores de Docker un mayor control sobre la memoria del sistema.
- Se corrigió un error para corregir el mecanismo de conexión cuando se intenta adjuntar desde un proceso de host a un proceso de Java que se encuentra en un contenedor de Docker.
- Nuevas API para permitir mejor la creación de colecciones no modificables. copyOf, Set.copyOf, y Map.copyOf son métodos que crean nuevas instancias de colección a partir de instancias existentes.
- toUnmodifiableList, toUnmodifiableSet, y toUnmodifiableMap se agregaron a la clase Collectors en el paquete Stream, lo que permite que los elementos de un Stream se recopilen en una colección no modificable.
- Una inferencia de tipo de variable local, para mejorar el lenguaje y extender la inferencia de tipo a las variables locales. La intención es reducir la "ceremonia" asociada con la codificación mientras se mantiene un compromiso con la seguridad de tipo estático.
- Una interfaz limpia del recolector de basura para mejorar el aislamiento del código fuente de diferentes recolectores de basura. Los objetivos de este esfuerzo incluyen un mejor modularidad para el código interno de recolección de basura en la máquina virtual HotSpot y facilitar la adición de un nuevo recolector de basura a HotSpot.

C. *Funciones y opciones eliminadas en java 10*

- eliminación de soporte para el uso del anticuado LookAndFeel: ya no es posible que las aplicaciones utilicen versiones antiguas o no compatibles del LookAndFeel. Algunas aplicaciones tales como Nimbus y Agua usó nombres de clases antiguos para instanciar JDK internal Swing LookAndFeel. Por ejemplo:

```
javax.swing.UIManager.setLookAndFeel
  ("com.sun.java.swing.plaf.nimbus.NimbusLookAndFeel");
javax.swing.UIManager.setLookAndFeel("apple.laf.AquaLookAndFeel");
```

Para aquellos usuarios que utilizaban estos métodos antiguos ahora se han establecidos nuevos comandos.

Para Nimbus:

```
UIManager.setLookAndFeel("javax.swing.plaf.nimbus.NimbusLookAndFeel");
```

Para Aqua:

```
UIManager.setLookAndFeel(UIManager.getSystemLookAndFeelClassName());
```

- La eliminación de los métodos Runtime.getLocalizedInputStream y getLocalizedOutputStream: eran parte de un mecanismo de internacionalización obsoleto y no tienen un uso conocido.
- La eliminación de DOM API común: com.sun.java.browser.plugin2.DOM y sun.plugin.dom.DOMObject ha sido removido. Las aplicaciones ahora pueden manipular DOM utilizando el método, Netscape.javascript.JSObject
- La eliminación de métodos y campos de securityManager anteriores a la versión 1.2:
  getInCheck(metodo)
  classDepth(metodo)
  classLoaderDepth(metodo)
  currentClassLoader(metodo)
  currentLoadedClass(metodo)
  inClass(metodo)
  inCLassLoasder(metodo)

  el método obsoleto checkMemberAccess ha sido remplazado por SecurityException si el usuario no ha sido concedido AllPermission.

- La eliminación de policyTool: la herramienta de seguridad policyTool ha sido removida de JDK.
- La eliminación de los doclet antiguos (JDK6 6, JDK 7, JDK 8 ERA): El antiguo doclet estándar, que genera contenido HTML ha sido suspendido por un reemplazo, y eliminado en esta versión.

D. *Metodos puestos en deprecate*

Java ha anunciado que ciertos métodos se han puesto en el modo de deprécate debido a que su uso es poco útil o su utilización puede provocar conflictos al tener poca compatibilidad con la nueva actualización.

- API Deprecate forRemoval: las clases deprecated java.security.(Certificate,Identity, IdentityScope, Signer) están sujetas para eliminación en futuras versiones en Java SE.
- La API javax.security.ac1 obsoleta se ha marcado forRemoval=true y está sujeta a eliminación en una versión futura de Java SE

IV. METODOLOGIA

A continuación, se ejecuto un ejemplo de los nuevos métodos de la aplicación de java 10. Se analizará el método de UnmodificableList la cual fue agregada a la clase de Collectores en el paquete Stream.

```
package Practica_1;

import java.util.Arrays;
import java.util.Collections;
import java.util.List;
import java.util.stream.Collectors;

/**
 *
 * @author Dutan2000
 */
public class Ejemplo {
    public static void main(String [] args){
        List<String> tech= Arrays.asList("Java", ".Net", "Spring", "Springboot", "Git", "REST");
        List<String> finalTechList =tech.stream().filter(k -> k.length()>3).collect(Collectors.toList());
        List<String> ListaInmodificable=Collections.unmodifiableList(finalTechList);
        //ListaInmodificable.add("nuevo");
    }
}
```

Como podemos observar se ha utilizado el método de Stream y su método de filter() y al final al momento de ejecutar el programa no se lanza ningún error y la aplicación se ejecuta con éxito. A continuación, se agregará un elemento a la lista y se observa que es lo que sucede.

```
package Practica_1;

import java.util.Arrays;
import java.util.Collections;
import java.util.List;
import java.util.stream.Collectors;

/**
 *
 * @author Dutan2000
 */
public class Ejemplo {
    public static void main(String [] args){
        List<String> tech= Arrays.asList("Java", ".Net", "Spring", "Springboot", "Git", "REST");
        List<String> finalTechList =tech.stream().filter(k -> k.length()>3).collect(Collectors.toList());
        List<String> ListaInmodificable=Collections.unmodifiableList(finalTechList);
        ListaInmodificable.add("nuevo");
    }
}
```

```
Exception in thread "main" java.lang.UnsupportedOperationException
        at java.util.Collections$UnmodifiableCollection.add(Collections.java:1055)
        at Practica_1.Ejemplo.main(Ejemplo.java:22)
/Users/newuser/Library/Caches/NetBeans/8.2/executor-snippets/run.xml:53: Java returned: 1
BUILD FAILED (total time: 0 seconds)
```

Al momento de querer modificar la lista o agregar un elemento, la aplicación de Java lanza una error, esto se debe a que el método de UnmodificableList() no permite la modificación de la lista existente.

A. *Conclusion*

En conclusión, podemos decir que la nueva actualización de Java se ha concentrado en mejor su sistema. Ha dedicado la mayoría de su tiempo a eliminar métodos y aplicaciones que no se utilizan muy frecuentemente. En cuanto ha la creación de nuevos métodos, se podría decir que solo se ha fijado en crear unos cuantos métodos para simplificar el proceso de codificación.

REFERENCIAS

[1] Arnab S. (17 de abril del año 2018). Java 10: New Features And Enhancements. www.dzone.com/articles/java-10-new-features-and-enhancements

[2] Krill P. (20 de marzo del año 2018). JDK 10: What's new in Java 10. www.infoworld.com/article/3230507/java-jdk-10-what-new-features-to-expect-in-the-next-java.html#:~:text=JDK%2010%2C%20an%20implementation%20of,to%20end%20in%20six%20months.