**Tarea 2**

**Programación-2**

**Patel, Ibrahim 8-920-710**

**Cuál es la última versión de Java 8?**

La última actualización de Java 8 es la actualización limitada *Java 8 Update 60* que fue lanzada el 18 de agosto de 2015.

**Cuáles son sus novedades?**

Las novedades principales de la versión Java 8 Update 60 (8u60) son:

Datos IANA 2015e

JDK 8u60 contiene datos de zona horaria IANA versión 2015e. Para obtener más información, consulte Versiones de datos de zona horaria en el software de JRE.

Corrección de bug: dns\_lookup\_realm debe estar definido en false por defecto

El valor de dns\_lookup\_realm del archivo Kerberos' krb5.conf es por defecto false. Consulte 8080637.

Corrección de bug: Desactivar los conjuntos de cifrado RC4

Los conjuntos de cifrado TLS basados en RC4 (por ejemplo, TLS\_RSA\_WITH\_RC4\_128\_SHA) ya no son seguros, por lo que no deben seguir usándose (consulte RFC 7465). Como consecuencia, los conjuntos de cifrado TLS basados en RC4 se han desactivado por defecto en la implantación de Oracle JSSE agregando "RC4" a la propiedad de seguridad "jdk.tls.disabledAlgorithms" y eliminándolos de la lista de conjuntos de cifrado activados por defecto. Estos conjuntos de cifrado pueden reactivarse eliminando "RC4" de la propiedad de seguridad "jdk.tls.disabledAlgorithms" en el archivo java.security o llamando de forma dinámica a Security.setProperty() y también volviendo a agregarlos a la lista de conjuntos de cifrado activados con los métodos SSLSocket/SSLEngine.setEnabledCipherSuites(). También puede utilizar la opción de línea de comando -Djava.security.properties para sobrescribir la propiedad de seguridad jdk.tls.disabledAlgorithms. Por ejemplo:

java -Djava.security.properties=my.java.security ...

donde my.java.security es un archivo que contiene la propiedad sin RC4:

jdk.tls.disabledAlgorithms=SSLv3

Incluso con esta opción definida desde la línea de comandos, deben volver a agregarse los conjuntos de cifrado basados en RC4 a la lista de conjuntos de cifrado activados mediante los métodos SSLSocket/SSLEngine.setEnabledCipherSuites(). Consulte 8076221.

Corrección de bug: Soportar la detección de tipo de almacén de claves para almacenes de claves JKS y PKCS12

Modo de compatibilidad de almacén de claves: Para mejorar la interoperabilidad, el tipo de almacén de claves de Java JKS ahora soporta el modo de compatibilidad de almacén de claves por defecto. Este modo permite a los almacenes de claves de JKS acceder a los formatos de archivo JKS y PKCS12. Para desactivar el modo de compatibilidad de almacén de claves, defina la propiedad de seguridad keystore.type.compat en el valor de cadena false. Consulte 8062552.

Corrección de bug: Métodos de supervisión Unsafe anticuados en la versión JDK 8u

Los métodos monitorEnter, monitorExit y tryMonitorEnter en sun.misc.Unsafe se han marcado como anticuados en JDK 8u60 y se eliminarán en una futura versión. Estos métodos no se utilizan en el JDK en sí y rara vez se usan fuera del JDK. Consulte 8069302.

Corrección de bug: Extraer la grabación de JFR del archivo principal mediante SA

DumpJFR es una herramienta basada en agente de capacidad de servicio que puede usarse para extraer datos de Java Flight Recorder (JFR) de los archivos principales y los procesos de Hotspot activos. Se puede usar DumpJFR mediante uno de los métodos siguientes:

Añada DumpJFR a un proceso activo:

java -cp $JAVA\_HOME/lib/sa-jdi.jar sun.jvm.hotspot.tools.DumpJFR <pid>

Añada DumpJFR a un archivo principal:

java -cp $JAVA\_HOME/lib/sa-jdi.jar sun.jvm.hotspot.tools.DumpJFR <java> <core>

La herramienta DumpJFR vuelca los datos de JFR en un archivo denominado recording.jfr en la carpeta de trabajo actual. Consulte 8065301 (no público).

Corrección de bug: Las variables locales denominadas 'enum' provocan fallos falsos del compilador

El analizador javac analiza de forma incorrecta variables locales con el nombre 'enum'; esto provoca fallos falsos cuando un programa con dichas variables locales se compila con un indicador 'source' que corresponde a una versión en la que la construcción enum no está disponible (como '-source 1.4'). Consulte 8069181.

Java Development Kit para ARM versión 8u60

Esta versión incluye Java Development Kit para ARM versión 8u60 (JDK 8u60 para ARM). Para obtener información de soporte de dispositivos ARM, consulte la página Descargas de JDK para ARM. Para obtener una lista de los requisitos del sistema, instrucciones de instalación y consejos de resolución de problemas, consulte la página Instrucciones de instalación.

Limitación: El soporte de Native Memory Tracking está limitado en JDK para ARM. La opción de línea de comando de Java XX:NativeMemoryTracking=detail no está soportada para destinos de ARM (se muestra un mensaje de error al usuario). En su lugar utilice la siguiente opción:

XX:NativeMemoryTracking=summary

Actualizaciones en la documentación por las mejoras de Nashorn

JDK 8u60 incluye nuevas mejoras de Nashorn. Como consecuencia, deben leerse los siguientes cambios en la documentación junto con la documentación de Nashorn existente:

Adición: En la sección anterior, mencionamos que todos los objetos JavaScript expuestos a API Java implantan la interfaz java.util.Map. Esto se aplica también a las matrices de JavaScript. Sin embargo, a menudo este no es el comportamiento que se espera o desea cuando el código Java espera objetos analizados por JSON. Las bibliotecas de Java que manipulan objetos analizados por JSON normalmente esperan matrices para exponer en su lugar la interfaz java.util.List. Si necesita exponer sus objetos de JavaScript para exponer las matrices como listas y no como mapas, puede utilizar la función Java.asJSONCompatible(obj), donde obj es la raíz del árbol de objetos de JSON.

Corrección: La precaución mencionada al final de la sección Asignación de tipos de datos ya no es aplicable. Nashorn garantiza que las cadenas de JavaScript internas se convierten a java.lang.String al exponerse de forma externa.

Corrección: La afirmación de la sección Asignación de tipos de datos, que menciona "Por ejemplo, las matrices deben convertirse de forma explícita,..." no es correcta. Las matrices se convierten de forma automática a tipos de matriz Java, como java.util.List, java.util.Collection, java.util.Queue y java.util.Deque, etc.

Cambios en el juego de reglas de despliegue v1.2

JDK 8u60 implanta el juego de reglas de despliegue (DRS) 1.2, que incluye los siguientes cambios:

Agregue el elemento "checksum" como subelemento de "id", lo que permite que los jar no firmados sean identificados por el total de control SHA-256 del formato sin comprimir de un jar:

El elemento "checksum" se hará coincidir solo con jar no firmados, y el hash proporcionado se comparará solo con el formato no comprimido del jar.

El elemento "checksum" (similar al elemento "certificate") tiene dos argumentos "hash" y "algorithm"; sin embargo, a diferencia de lo que ocurre con el elemento "certificate", el único valor soportado para "algorithm" es "SHA-256". Se ignorará cualquier otro valor proporcionado.

Permite que el elemento "message" se aplique a todos los tipos de regla, donde antes solo se aplicaba a una regla de bloque:

En una regla de ejecución, un subelemento de mensaje hará que se muestre un cuadro de diálogo de mensaje mientras que, sin una regla de ejecución, el comportamiento por defecto sería mostrar un cuadro de diálogo de certificado o formato no firmado. El mensaje se mostrará en el cuadro de diálogo de mensaje.

En una regla por defecto, el mensaje solo se mostrará si la acción por defecto es el bloqueo. En ese caso, el mensaje se incluirá en el cuadro de diálogo de bloque.

Repita los bloques "customer" en la consola de Java, los archivos de rastreo y los registros de Java Usage Tracker.

Antes de la versión DRS 1.2, los elementos "customer" podían incluirse (con cualquier subelemento) en el archivo ruleset.xml. Tanto este elemento como todos sus subelementos se ignoran. En DRS 1.2, los elementos siguen ignorándose desde el punto de vista funcional. No obstante:

Al analizar el archivo ruleset.xml, todos los bloques "customer" se repetirán en la consola de Java y el archivo de rastreo de despliegue (si la consola y el rastreo están activados).

Al utilizar una regla, todos los registros "customer" incluidos en dicha regla se agregarán al registro de Java Usage Tracker (JUT) (si JUT está activado).

**Cuando se lanzara la próxima versión de Java?**

Fecha de caducidad de Java

La fecha de caducidad de 8u60 es el 20 de octubre de 2015. Java caduca cada vez que hay disponible una nueva versión con correcciones a las vulnerabilidades de seguridad. Para los sistemas que no se pueden ejecutar en servidores Oracle, un mecanismo secundario se encargará de caducar esta versión de JRE (versión 8u60) el 20 de noviembre de 2015. Una vez se haya cumplido cualquiera de las condiciones (la nueva versión esté disponible o se haya alcanzado la fecha de caducidad) Java enviará mensajes de advertencia y recordatorios sobre la nueva versión.