26/12/23

Java Features

* Java 1

1) AWT Event Model [Abstract Window Toolkit]
- It is an event-handling model, enabling developers to respond to user interactions like button clicks or mouse movements in graphical user interfaces.

2) Inner Classes
- It is allowing the definition of a class within another class & it enhances encapsulation & code organization.

3) Java Beans
- It was introduced for reusable software components.
- These are Java classes adhering to specific ~~comp~~ conventions, promoting the development of modular and customizable applications.

4) JDBC [Java Database Connectivity]
- It provides a standard interface for Java application to interact with relational databases, allowing for database connectivity and data manipulation.

5) RMI [Remote Method Invocation]
- It is enabling distributed computing by allowing objects to invoke methods on objects in different Java Virtual Machines (JVMs) across a network.

6) Reflection & Introspection
- Reflection, supporting introspection in Java 1, allowed examination of class information at runtime. However, modifications to classes at runtime were not possible.

7) JIT Compiler for Windows
- It is enhancing the runtime performance of Java applications by translating Java bytecode into native machine code at runtime.

* JAVA 2
1) strictfp Keyword
- It is ensuring consistent floating-point calculations across different platforms, promoting platform-independent numerical results.

2) Swing Graphical API
- It is a powerful and flexible graphical user interface (GUI) toolkit, providing a more sophisticated and customizable alternative to the original AWT Components.

3) JIT Compiler for Sun's JVM
- It was equipped with a Just-In-Time (JIT) compiler for the first time, improving the runtime performance of Java programs by translating bytecode into native machine code on-the-fly

4) Java Plug-in
- The concept of java plug-ins was introduced, allowing web browsers to run Java applets seamlessly, enhancing the ~~interaction~~ integration of Java applications within web pages.

5) Collections Framework
- It provides a comprehensive set of interfaces and classes for handling and manipulating collections of objects, offering a more organized and effecient approach to data storage & retrieval

* JAVA 3
1) HotSpot JVM
- It is an advanced virtual machine with adaptive optimization techniques, significantly improving Java application performance through dynamic runtime optimizations.

2) JNDI [Java Naming & Directory Interface]
- JNDI provided a standard interface for accessing and managing naming and directory services, enabling Java applications to interact with various naming and directory systems

3) Java Platform Debugger Architecture [JPDA]
- A framework that facilitates debugging of Java applications at both local and remote levels, enhancing the debugging capabilites for developers

4) JavaSound
- A comprehensive API for handling audio functionality in Java applications, allowing developers to incorporate Sound and music into their programs

5) Synthetic Proxy Classes
- Enabling dynamic generation of proxy classes at runtime which is particularly useful for implementing aspects of aspect-oriented programming & other dynamic code generation scenarios

* JAVA 4

1) assert Keyword
- It use in code to simplify testing and debugging by allowing developers to embed assertions directly into their programs.

2) Regular Expressions
- It incorporated regular expressions, providing a powerful and flexible pattern matching mechanism for string manipulation and searching within the 'java.util.regex' package.

3) Exception Chaining
- It is allowing exceptions to be nested within one another, providing more detailed information about the sequence of errors that occured.

4) IPv6 Support
- It ensures compatibility with the evolving internet infrastructure.

5) New I/o [NIO]
- It is offering a scalable I/o framework with improved performance and non-blocking capabilities for handling large-scale data.

6) Logging API
- It is providing a standard way for Java applications to generate ~~way~~ for Java applications to log messages, facilitating better management and analysis of application logs

7) Image I/o API
- It is offering a standardized approach to read and write images in various formats, enhancing image processing capabilities in Java applications.

8) Integrated XML Parser and XSLT Processor [JAXP]
- It is for XML Processing (JAXP), providing a standard interface for processing XML documents including parsing and transforming using XSLT.

9) Integrated Security and Cryptography Extensions (JCE, JSSE, JAAS)
- It is ~~tnt~~ for enhanced security features, supporting encryption, authentication and access control.

10) Java Web Start
- It is allowing users to launch Java applications directly from the web without the need for manual installation, simplifying the deployment for Java applications.

11) Preferences API (java.util.prefs)
- It is providing a cross-platform and persistent storage mechanism for application preferences and configuration settings.

* JAVA 5

1) Generics
- Enabling developers to create more flexible and type-safe collections and classes by allowing the use of paramaterized types.

2) Annotations
- It provides a metadata facility to add information and behaviors to Java code, improving code organization and enabling tools to process metadata.

3) Autoboxing/Unboxing
- Simplifying the conversion between primitive types and their corresponding wrapper classes

4) Enumerations
- Providing a type-safe and concise way to define sets of constants, enhancing code readability and reducing error-prone practices.

5) Varargs
- Allowing methods to accept a variable number of arguments, simplifying method invocation with a variable number of parameters.

6) Enhanced for-each loops
- Simplifying the interration over collections and arrays, improving code readability

7) Static Imports
- Allowing static members of a class to be imported directly, reducing verbosity and improving code readability

8) New Concurrency Utilities in java.util.concurrent
- Providing a higher-level framework for concurrent programming, including Executors, concurrent collections and the Fork/Join framework.

9) Scanner Class
- Simplifying the parsing of data from various input streams and buffers, enhancing the capabilities for interactive user input and file parsing

* JAVA 6

1) Scripting Language Support
- It support with the Inclusion of the Java Compiler API, allowing developers to seamlessly integrate dynamic languages like JavaScript and Ruby into Java apps.

2) Performance Improvements
- It improves startup time, reduced memory footprints, and optimizations in the Hotspot JVM, contributing to overall better runtime performance.

3) JAX-WS [Java API for XML Web Services]
- It is providing a standardized approach to developing web services, making it easier to create, deploy, and consume web services in Java

4) JDBC 4
- It is offering enhancements in database connectivity, including automatic driver loading, improved exception handling, and support for SQL XML.

5) Java Compiler API
- It is enabling dynamic compilation of Java Source code within Java programs, allowing developers to generate and compile code at runtime

6) JAXB 2.0 and StAX Parser
- JAXB 2.0 for XML data binding and the StAX parser for streaming XML processing. improving XML handling capabilities in Java apps

7) Pluggable Annotation
- Allowing developers to create custom annotation processors and tools to analyze and generate code based on annotations in a modular & extensible manner.

8) New GC Algorithms
- Including the Garbage-First (G1) Collector, offering improved garbage collection performance and better adaptability to different application server scenarios

* Java 7
1) JVM Support for Dynamic Languages
- Improving performances and compatibility for Languages like Groovy and JRuby.

2) Compressed 64-bit Pointers
- Reducing memory overhead on 64-bit systems, and improving performance and efficiency

3) Strings in Switch
- Providing a more expressive and readable way to handle multiple conditions based on string values

4) Automatic Resource Management (try-with-resource)
- Simplifying resource management by automatically closing resources like files or sockets at the end of a try statement.

5) Diamond Operator
- Allowing the compiler to infer the generic type parameters, reducing verbosity when creating instances of generic classes

6) Simplified Varargs method declaration
- Allowing the use of non-reifiable types (e.g List<String>), improving flexibility in method signatures.

7) Binary Integer Literals
- Allowing developers to specify integral values using the binary format (e.g., '0b1101' for the decimal 13)

8) Underscores in Numeric Literals
- Allowed the use of underscores in numeric literals, improving readability (e.g., '1_000_000' instead of '1000000')

9) Improved Exception Handling
- Introduced multi-catch and improved rethrowing of exceptions, simplifying exception handling & making code more concise.

10) ForkJoin Framework
- Providing a framework for parallel programming, especially suited for recursive algorithms and divide-and-conquer tasks

11) NIO 2.0
- Introducing support for multiple file systems, file metadata, symbolic links, and the WatchService for monitoring file system changes.

12) TimeSort for Sorting
- Default sorting algorithm for collections and arrays of objects, offering improved performance and adaptability compared to the previous merge sort

13) APIs for Graphics Features
- Enhancing support for rendering and manipulating graphical elements in Java apps

14) Support for New Network Protocols
- Including SCTP and Sockets Direct Protocols, expanding networking capabilities.

* JAVA 8
1) Lambda Expression Support in APIs
- Enabling a concise way to express instances of single-method interfaces, enhancing code readability and supporting functional programming paradigms APIs

2) Stream API
- Providing functional approach to process sequences of elements, facilating parallelism and simplifying complex data manipulations.

3) Functional Interface and Default Methods
- Allowing interfaces to have a single abstract method, and default methods, enabling the addition of methods to interfaces without breaking existing implementations

4) Optionals
- Providing a more expressive and null-safe way to represent optional values, reducing the likelihood of null pointer exceptions

5) Nashorn - JavaScript Runtime
- A lightweight and high-performance Javascript runtime, allowing developers to embed and execute Javascript Code within Java applications

6) Annotation on Java Types
- Including type declarations, providing more flexibility in expressing metadata.

7) Unsigned Integer Arithmetic
- Providing methods for performing arithmetic operations on unsigned integer values

8) Repeating Annotations
- Allowing multiple annotations of the same type to be applied to a declaration, enhancing flexibility and reducing code verbosity

9) New Date and Time API
- Addressing the shortcomings of the existing java.util.Date and java.util.Calender classes and

providing a more modern and flexible approach to date and time manipulation.

10) Statically - Linked JNI Libraries
- Improving performance and simplifying deployment by eliminating the need for separate dynamic loading

11) Launch JavaFX Apps from JAR files
- Streamlining the deployment of JavaFX apps

12) Remove the Permanent Generation from GC
- Providing more flexibility for storing metadata related to class definitions and reducing the likelihood of OutofMemoryError due to class loading

* JAVA 9
1.) Java Platform Module System
- Providing a modular structure for the JDK and allowing developers to create more modular & scalable apps

2) Interface Private Methods
- Enabling developers to provide shared code within an interface without exposing it to external classes

3) HTTP 2 Client
- Providing a modern and more efficient API for handling communication

4) JShell - REPL Tool
- Read-Eval-Print Loop tool, allowing developers to interactively experiment with Java code, test snippets, and learn the language more dynamically

5) Platform and JVM Logging
- Standardized logging API for the platform & JVM, improving logging consistency across Java apps

6) Process API Updates
- Providing better control over native processes, including the ability to handle and control process streams

7) Collection API Updates
- It includes convenience factory methods for creating immutable collections

8) Multi-Release JAR files
- Allowing developers to include different versions of classes for different Java runtime versions within a single JAR.

9) @Deprecated Tag Changes
- Providing more information about deprecation and allowing for improved documentation

10) Stack Walking
- Facilitating more efficient and fine-grained access to stack frames for improved debugging & profilin

11.) JavaDocs Updates
- Providing better documentation & improved support for searching & accessing information

* JAVA 10

1) JEP 286 : Local Variable Type Inference
- Allowing more concise code while retaining Static typing.

2) JEP 322 : Time - Based Release Versioning
- Moving away from the feature-driven release model to a time-driven model for more predictable and regular Java releases.

3) JEP 304 : Garbage - Collector Interface
- Making it easier to develop and plug in new garbage collectors.

4) JEP 307 : Parallel Full GC for G1
- Enhanced the Garbage-First Collector with parallel full garbage collection, improving the efficiency of G1 for certain workloads

5) JEP 316 : Heap Allocation on Alternative Memory Devices
- The ability to allocate the java object heap on alternative memory devices, improving flexibility in memory management.

6) JEP 296 : Consolidate the JDK Forest into a Single Repository
- Consolidated the JDK source code into a single repository, simplifying the development and maintenance of the Java platform

7) JEP 310 : Application Class-Data Sharing
- Allowing the sharing of pre-compiled class data among multiple Java processes for faster startup times.

8) JEP 314 : Additional Unicode Language-Tag Extension
- Improving internationalization capabilities

9.) JEP 319 : Root Certificates
- Updated the set of root certificates for the default truststore, addressing security and certificate-related issues.

10) JEP 317 : Experimental Java-Based JIT Compiler (Graal)
- Graal providing an alternative to the existing HotSpot Compiler.

11.) JEP 312 : Thread-Local Handshakes
- Allowing more efficient communication between Java threads.

12.) JEP 313 : Remove the Native-Header Generation Tool
- Encouraging the use of the more modern and flexible javac options for native-header generation

13) New Added APIs and Options
- Java 10 included various new APIs and options, providing additional functionalities and Configurations

14) Removed APIs and Options
- Streamlining the platform and encouraging the use of more modern alternatives.

* Java 11

1) HTTP Client API
- Providing a standardized and more modern way to Send HTTP requests and handle responses.

2) Launch Single-file Programs Without Compilation
- Ability to launch single-file programs directly without explicit compilation, simplifying the execution of simple Java programs

3) String API Changes
- Introducing new methods for checking and transforming strings, improving convenience and readability

4) Collection.toArray (IntFunction)
- Allowing developers to create arrays with a specific element type, enhancing type safety

5) Files.readString() and Files.writeString()
- Providing more convenient ways to read and write content to files

6.) Optional.isEmpty()
- Offering a more explicit way to check whether an optional instance is empty.

* Java 12

1) Collectors.teeing () in Stream API
- A new collector in the stream API that allows developers to perform two collectors in parallel & Combine their results.

2) String API Changes
- Introducing new methods and improving the function-ality for better string manipulation

3) Files.mismatch (Path, Path)
- Allowing developers to efficiently find the index of the first differing byte into two files.

4) Compact Number formatting
- Providing a more concise and readable representation of large numbers

5) Support for Unicode 11
- Incorporating the latest Unicode standard for improved character handling and representation.

6.) Switch Expression
- Providing a more expressive and concise way to handle multiple conditions within a switch statement