**Diferencias Java VS Kotlin**

|  |  |  |  |
| --- | --- | --- | --- |
| **Sr. No.** | **Características** | **Kotlin** | **Java** |
| 1 | Características de expansión | Disponible en Kotlin. Las extensiones son útiles porque permiten a los desarrolladores agregar métodos a las clases sin realizar cambios en el código fuente. | En Java, necesitamos crear una nueva clase para extender la funcionalidad de la clase existente, por lo que a partir de ahora, los usuarios pueden usar una clase recién creada en cualquier lugar para usar la funcionalidad extendida. |
| 2 | Seguridad cero | Disponible. El tipo de sistema de Kotlin tiene seguridad cero incorporada. | No disponible. De hecho, NullPointerException es el principal responsable de los errores de desarrollo de Android y Java. |
| 3 | Miembros estáticos | En Kotlin, podemos usar el objeto complementario para crear miembros estáticos de la clase. Una declaración de objeto en una clase se puede etiquetar con la palabra compañera. | Disponible en Java. Se utiliza principalmente para la gestión de la memoria. Puede usar palabras clave estáticas de Java con variables, métodos, bloques y clases anidadas. |
| 4 | Pintores de cuerdas | Sí, hay dos tipos de literales de cadena en Kotlin que escaparon de la cadena y la cadena sin procesar. La plantilla de cadena de Kotlin también admite la expresión. | Disponible también en Java, pero no admite expresiones como Kotlin. |
| 5 | Corutinas | En Kotlin, las corutinas son patrones de diseño simultáneamente. Se puede usar para simplificar el código en Android que funciona de forma asíncrona. Coroutines se agregó a Kotlin en la versión 1.3 y se basan en conceptos establecidos de otros idiomas. | En Java, los usuarios tienen dos opciones diferentes, como RxJava y Project Loom. RxJava es una biblioteca para componer aplicaciones asíncronas y basadas en eventos que utilizan secuencias observables, y Project Loom admite un modelo de concurrencia ligero y de alto rendimiento en Java. |
| 6 | Tipos de comodines | No, Kotlin d no tiene ningún tipo de comodín. Pero tiene otras dos cosas, incluida la varianza del sitio y las proyecciones de tipo. | Disponible en Java. Los comodines en código general significan (?) Representando un tipo desconocido. Se puede usar en diferentes situaciones. |
| 7 | Elenco inteligente | Sí, esta función está disponible en Kotlin. Ayuda al compilador de Kotlin a rastrear condiciones dentro de cuya expresión. Si el compilador encuentra una variable que no es nula del tipo anulable, el compilador da acceso a la variable. | No, no está disponible en Java. Sin embargo, para conocer los tipos en Java, podemos usar este ejemplo para verificar el tipo y luego lanzarlo al tipo correcto. |
| 8 | Carga diferida | Sí, está disponible en Kotlin. Principalmente redujo el tiempo de inicio que es útil cuando se usa Kotlin para el desarrollo de aplicaciones. | Esta característica no está disponible en Java. |
| 9 | Sin excepciones controladas | Kotlin eliminó las excepciones por completo para minimizar la verbosidad y mejorar la seguridad de los tipos. | En Java, la función de excepción marcada es un problema que causa bloques de captura vacíos. |
| 10 | Sobrecarga del operador | Sí, Kotlin permite a los usuarios ofrecer una forma de llamar a las funciones. Permite realizar una operación aritmética, verificaciones de similitud o comparación de qué tipo de objeto queremos, mediante símbolos como +, -, /, \*,%, <, >. | En Java, los operadores están asociados con ciertos tipos de Java. Por ejemplo, los tipos de cadena y numéricos en Java pueden usar el operador + para concatenación y adiciones. Otro tipo de Java no puede reutilizar este operador. |

**¿Por qué es mejor Kotlin?**

Kotlin es perfecto para desarrollar aplicaciones del lado del servidor, permitiendo a los usuarios escribir código más conciso que Java, para que puedan resolver los mismos problemas con menos códigos. Mejora la capacidad de mantenimiento y la legibilidad del código, lo que significa que los ingenieros pueden escribir, leer y cambiar el código de manera más eficiente.

En resumen, Kotlin se puede usar para todo lo que Java se usa hoy en día, incluido el desarrollo de servidores, clientes, web y Android. Estas son algunas de las otras razones para elegir Kotlin en lugar de Java:

* Compilador más inteligente y seguro
* Más fácil de mantener
* Aumento de la productividad
* Código más seguro
* Eliminación de referencias cero