APRENDERAPROGRAMAR.COM

VERSIONES Y DISTRIBUCIONES JAVA: J2SE, J2EE, J2ME. ¿JAVA 6, JAVA 7, JAVA 8, JAVA 9… CUÁL ES MEJOR USAR? (CU00606B)

Sección: Cursos
Categoría: Curso “Aprender programación Java desde cero”

Fecha revisión: 2029

Resumen: Entrega nº6 curso Aprender programación Java desde cero.

Autor: Alex Rodríguez

VERSIONES Y DISTRIBUCIONES DE JAVA

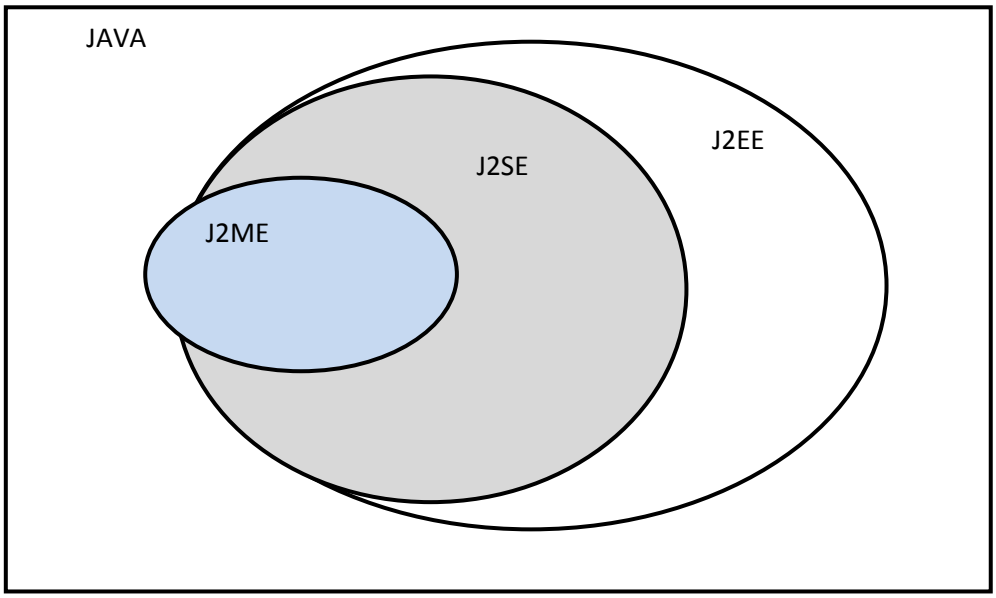
Java, como la mayoría de los lenguajes, ha sufrido diversos cambios a lo largo de su historia. Además, en cada momento han coexistido distintas versiones o distribuciones de Java con distintos fines. Actualmente puede considerarse que el Java vigente se denomina Java 2 y existen 3 distribuciones principales de Java 2, con ciertos aspectos comunes y ciertos aspectos divergentes.

Estas tres distribuciones son:

a) **J2SE o simplemente Java SE**: Java 2 Standard Edition o Java Standard Edition. Orientado al desarrollo de aplicaciones cliente / servidor. No incluye soporte a tecnologías para internet. Es la base para las otras distribuciones Java y es la plataforma que utilizaremos nosotros en este curso por ser la más utilizada.

b) **J2EE:** Java 2 Enterprise Edition. Orientado a empresas y a la integración entre sistemas. Incluye soporte a tecnologías para internet. Su base es J2SE.

c) **J2ME**: Java 2 Micro Edition. Orientado a pequeños dispositivos móviles (teléfonos, tabletas, etc.).



En esta imagen vemos, de forma orientativa, como J2EE “expande” a J2SE, mientras que J2ME “recorta” a J2SE al tiempo que tiene una fracción de contenido diferenciada exclusiva de J2ME. En realidad hablar de expansiones y recortes no es correcto, porque cada distribución es en sí misma distinta puesto que están concebidas con distintas finalidades. Por tanto no puede decirse que sean expansiones o recortes, pero de forma coloquial muchas veces se interpreta así.

Java hoy en día es más que un lenguaje de programación, como veremos más adelante. El lenguaje Java estándar ha experimentado numerosos cambios desde la versión primigenia, JDK 1.0, así como un enorme incremento en el número de recursos disponibles para los programadores Java. Podemos citar en la evolución del Java estándar:

- **JDK 1.0** (1996): primer lanzamiento del lenguaje Java.
- **JDK 1.1** (1997): mejora de la versión anterior.
- **J2SE 1.2** (1998): ésta y las siguientes versiones fueron recogidas bajo la denominación Java 2 y el nombre "J2SE" (Java 2 Platform, Standard Edition), reemplazó a JDK para distinguir la plataforma base de J2EE (Java 2 Platform, Enterprise Edition) y J2ME (Java 2 Platform, Micro Edition). Incluyó distintas mejoras.
- **J2SE 1.3** (2000): mejora de la versión anterior.
- **J2SE 1.4** (2002): mejora de la versión anterior.
- **J2SE 5.0** (2004): originalmente numerada 1.5, esta notación aún es usada en ocasiones. Mejora de la versión anterior.
- **Java SE 6** (2006): en esta versión, Sun cambió el nombre "J2SE" por Java SE y eliminó el ".0" del número de versión. Mejora de la versión anterior.
- **Java SE 7** (2011): nueva versión que mejora la anterior. Incluyó mayor soporte para XML.
- **Java SE 8** (2014): nueva versión que mejora la anterior. Incluye la posibilidad de embeber JavaScript con Java y mejoras en la gestión de fechas y tiempo.
- **Java SE 9**: nueva versión que mejora la anterior (en difusión).
- **Java SE 10**: nueva versión que mejora la anterior (todavía sin uso comercial).

En Java todas las versiones siguen los mismos estándares de datos, esto permite que un programa que hayamos hecho con una versión antigua, pueda ser ejecutado con una versión más nueva sin necesidad de ningún cambio.

Además de los cambios en el lenguaje en sí, con el paso de los años los recursos disponibles para los programadores Java que ofrece la empresa que desarrolla el lenguaje (antiguamente Sun MicroSystems, actualmente Oracle) han crecido enormemente. La denominada “biblioteca de clases de Java” (Java class library) ha pasado de ofrecer unos pocos cientos de clases en JDK 1.0 hasta cerca de 6000 en Java SE 8. Se han introducido recursos completamente nuevos, como Swing y Java2D, mientras que muchos de los métodos y clases originales de JDK 1.0 han dejado de utilizarse.

Cuando trabajamos con Java será frecuente que busquemos información “oficial” en internet. Cuando decimos oficial nos referimos a la que ofrece la propia empresa desarrolladora de Java. Cuando buscamos información sobre Java hay que tener cuidado respecto a a qué versión hace alusión la información. Por ejemplo, prueba a buscar “ArrayList java” o “ArrayList api java” en google, yahoo, bing o cualquier otro buscador. Un resultado posible es el siguiente (fíjate que en un caso es Java 1.4 y en otro Java SE 7):

- ArrayList (Java 2 **Platform SE v1.4.2**)

java.util. Class *ArrayList*. *java*.lang.Object extended by *java*.util.AbstractCollection extended by *java*.util.AbstractList extended by **...**
download.oracle.com/javase/.../java/.../ArrayList.html - En caché - Similares

- ArrayList (Java **Platform SE 7**)

java.lang.Object extended by *java*.util.AbstractCollection<E> extended by **...**
download.oracle.com/javase/7/.../java/.../ArrayList.html - En caché - Similares

Nosotros en este curso trabajaremos con Java Platform SE 6 (Standard Edition) o Java SE 7 por ser las versiones más usadas hoy en día: si miramos la documentación correspondiente a versiones anteriores podemos confundirnos. Los ejemplos que mostramos en el curso son de Java SE 7. Por tanto una búsqueda más correcta sería "ArrayList api java 7", y en todo caso **estar atentos a la especificación de la documentación** para comprobar que efectivamente se corresponde con la versión con la que estemos trabajando. Si quieres utilizar otra versión Java no hay problema siempre que sea versión 6 o superior. Los cambios entre versiones no suelen ser tan importantes como para afectar a una persona que aprende el lenguaje por primera vez: en realidad nos daría igual usar una versión u otra. Sin embargo, hay que tener claro qué versión es la que usamos.

Hemos usado el término api en las búsquedas: estas siglas corresponden a "Application Programming Interface" o interfaz de programación de aplicaciones. De momento, pensar que API equivale a "recursos" que nos ofrece el lenguaje Java (o si se prefiere, recursos que nos ofrece la empresa que lo desarrolla) para crear aplicaciones. Por ejemplo, podemos pretender ordenar una lista de números denominada Lista1. Podemos hacerlo de dos maneras: escribir las instrucciones paso a paso para que tenga lugar la ordenación, o usar un recurso ya disponible (algo así como "Lista1.usarRecursoOrdenar"). A medida que vayamos avanzando, nos iremos familiarizando poco a poco con el API de Java.

Próxima entrega: CU00607B

Acceso al curso completo en aprenderaprogramar.com -- > Cursos, o en la dirección siguiente:

http://www.aprenderaprogramar.com/index.php?option=com_content&view=category&id=68&Itemid=188