¿Qué elemento es imprescindible para que una aplicación escrita en Java pueda ejecutarse en un ordenador?

1. Que disponga de conexión a Internet y del hardware adecuado.
2. Que tenga instalado un navegador web y conexión a Internet.
3. Que tenga la Máquina Virtual Java adecuada instalada.

Junto a las características diferenciadoras del lenguaje Java relacionadas con la independencia y el trabajo en red, han de destacarse dos virtudes que hacen a este lenguaje uno de los más extendidos entre la comunidad de programadores: su **seguridad** y su **simplicidad**.

- **Seguridad:** en primer lugar, los posibles accesos a zonas de memoria "sensibles" que en otros lenguajes como C y C++ podían suponer peligros importantes, se han eliminado en Java.

  Y en segundo lugar, el código Java se comprueba y verifica para evitar que determinadas secciones del código produzcan efectos no deseados. Los test que se aplican garantizan que las operaciones, operandos, conversiones, uso de clases y demás acciones son seguras.

  En definitiva, podemos afirmar que Java es un lenguaje seguro.

- **Simplicidad:** aunque Java es tan potente como C o C++, es bastante más sencillo. Posee una curva de aprendizaje muy rápida y, **para alguien que comienza a programar en este lenguaje,** como será el caso de la mayoría de quienes comienzan a estudiar este módulo, le resulta relativamente fácil comenzar a escribir aplicaciones interesantes.

  Si has programado alguna vez en C o C++ encontrarás que Java te pone las cosas más fáciles, ya que se han eliminado: la aritmética de punteros, los registros, la definición de tipos, la gestión de memoria, etc. Con esta simplificación se reduce bastante la posibilidad de cometer errores comunes en los programas. Un programador experimentado en C o C++ puede cambiar a este lenguaje rápidamente y obtener resultados en muy poco espacio de tiempo.

  Muy relacionado con la simplicidad que aporta Java está la incorporación de un elemento muy útil como es el **Recolector de Basura (Garbage collector)**. Permite al programador liberarse de la gestión de la memoria y hace que ciertos bloques de memoria puedan reaprovecharse, disminuyendo el número de huecos libres (fragmentación de memoria).

  Cuando realicemos programas, crearemos objetos, haremos que éstos interaccionen, etc. Todas estas operaciones requieren de uso de memoria del sistema, pero la gestión de ésta será realizada de manera transparente al programador. Todo lo contrario que ocurría en otros lenguajes. Podremos crear tantos objetos como solicitemos, pero nunca

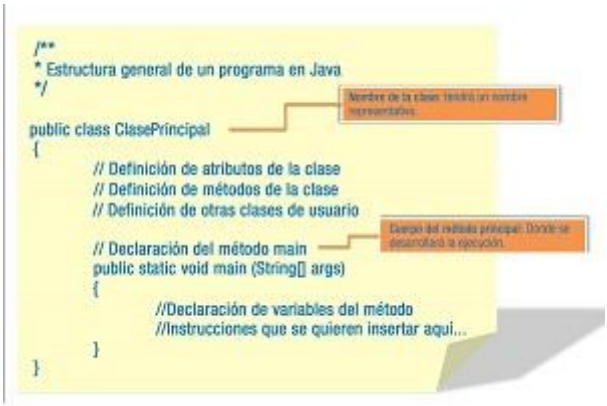
tendremos que destruirlos. El entorno de Java borrará los objetos cuando determine que no se van a utilizar más. Este proceso es conocido como **recolección de basura**, y simplifica tu trabajo al programar una barbaridad.

Rellena los huecos con los conceptos adecuados:
En Java se ha simplificado la gestión de memoria a través de la eliminación de la Aritmética de … , por lo que la incorporación del Garbage Collector evita que se produzca un crecimiento de los huecos libres en memoria, que recibe el nombre de… de memoria.

En Java el código fuente es compilado, obteniéndose el código binario en forma de bytecodes. Pero, ¿cuál es la extensión del archivo resultante?

1. Extensión .obj
2. Extensión .class
3. Extensión .java

En el gráfico al que puedes acceder a continuación, se presenta la estructura general de un programa realizado en un lenguaje orientado a objetos como es Java.



José Luis García Martínez. Uso educativo-nc. Elaboración propia.

Vamos a analizar los elementos que aparecen en esa estructura:

- public class ClasePrincipal: todos los programas han de incluir elemento como éste. Podrá llamarse ClasePrincipal, **ProgramaPrincipal**, ClaseDePruebas, **ProgramaDePruebas**, Programa01 o como queramos. Pero debe tener un nombre. Se trata de una clase general en la que se incluyen todos los demás elementos del programa. En unidades posteriores veremos qué es una clase y cuáles son sus componentes principales. Por ahora es suficiente con que sepamos que nuestro programa va a comenzar con las líneas public class NombrePrograma, donde NombrePrograma será el nombre de nuestro programa.
- Dentro de este elemento principal observamos el método o función **main()** que contiene las líneas de código de nuestro programa. También veremos más adelante qué es un método. Baste por ahora saber que, al igual que en el caso anterior, nuestro programa sebe contener también esas líneas public static void

main (String args[]). Aquí dentro se podrán incluir las instrucciones que estimemos oportunas para la ejecución del programa.

Ten en cuenta que **Java distingue entre mayúsculas y minúsculas**. Si le das a la clase principal el nombre PrimerPrograma, el archivo **.java** tendrá como identificador **PrimerPrograma.java**, que es totalmente diferente a primerprograma.java. Además, para Java los elementos PrimerPrograma y **primerprograma** serían considerados dos clases diferentes dentro del código fuente.

- **Comentarios**: los comentarios se suelen incluir en el código fuente para realizar aclaraciones, anotaciones o cualquier otra indicación que el programador estime oportuna. Estos comentarios pueden introducirse de dos formas:
  - Con **//** estaríamos estableciendo una única línea completa de comentario, es decir, todo lo que hay detrás, hasta que haya un salto de línea, es comentario.
  - Con **/* */**. De esta forma con **/*** comenzaríamos el comentario y éste no terminaría hasta que no insertáramos ***/**.
- **Bloques de código**: son conjuntos de instrucciones que se marcan mediante la apertura y cierre de llaves **{ }**. El código así marcado es considerado interno al bloque.
- **Punto y coma** (**;**): aunque en el ejemplo de la imagen no hemos terminado ninguna línea de código con punto y coma, para no distraer de momento con los detalles, hay que hacer hincapié en que cada línea de código ha de terminar con punto y coma (**;**). En caso de no hacerlo, tendremos errores sintácticos.

public static void main (String[] args) contiene las líneas de código de nuestro programa principal. ¿Verdadero o Falso?

Podemos desarrollar programas escritos en Java mediante un editor de textos y a través del JDK podremos ejecutarlos.

¿Verdadero o Falso?

Junto con el kit de desarrollo que hemos descargado e instalado anteriormente, vienen incluidas gratuitamente todas las bibliotecas de la **API** (Application Programming Interface – Interfaz de programación de aplicaciones) de Java, es lo que se conoce como **Bibliotecas de Clases Java.** Este conjunto de bibliotecas proporciona al programador paquetes de clases útiles para la realización de múltiples tareas dentro de un programa. Está organizada en paquetes lógicos, donde cada paquete contiene un conjunto de clases relacionadas semánticamente.

En décadas pasadas una biblioteca era un conjunto de programas que contenían cientos de rutinas (una rutina es un procedimiento o función bien verificados, en determinado lenguaje de programación). Las rutinas de biblioteca manejaban las tareas que todos o casi todos los programas necesitaban. El programador podía recurrir a esta biblioteca para desarrollar programas con rapidez.

Una biblioteca de clases es un conjunto de clases de programación orientada a objetos. Esas clases contienen métodos que son útiles para los programadores.

En el caso de Java cuando descargamos el JDK obtenemos la biblioteca de clases API. Utilizar las clases y métodos de las API de Java reduce el tiempo de desarrollo de los programas. También, existen diversas bibliotecas de clases desarrolladas por terceros que contienen componentes reutilizables de software, y están disponibles a través de la Web.

Indica qué no es la API de Java:

1. Un entorno integrado de desarrollo.
2. Un conjunto de bibliotecas de clases.
3. Una parte del JDK, incluido en el Java SE.

La versatilidad del lenguaje de programación Java permite al programador crear distintos tipos de aplicaciones. A continuación, describiremos las características más relevantes de cada uno de ellos:

- **Aplicaciones de consola:**
  - Son programas independientes al igual que los creados con los lenguajes tradicionales.
  - Se componen como mínimo de un archivo **.class** que debe contar necesariamente con el método **main()**.
  - No necesitan un navegador web y se ejecutan cuando invocamos el comando **java** para iniciar la Máquina Virtual de Java (JVM). De no encontrarse el método **main()** la aplicación no podrá ejecutarse.
  - Las aplicaciones de consola leen y escriben hacia y desde la entrada y salida estándar, sin ninguna interfaz gráfica de usuario.

- **Aplicaciones gráficas:**
  - Aquellas que utilizan las clases con capacidades gráficas, como Swing, que es la biblioteca para la interfaz gráfica de usuario avanzada de la plataforma Java SE.
  - Incluyen las instrucciones import, que indican al compilador de Java que las clases del paquete javax.swing se incluyan en la compilación.

- **Applets:**
  - Son programas incrustados en otras aplicaciones, normalmente una página web que se muestra en un navegador. Cuando el navegador carga una web que contiene un applet, éste se descarga en el navegador web y comienza a ejecutarse. Esto nos permite crear programas que cualquier usuario puede ejecutar con tan solo cargar la página web en su navegador.
  - Se pueden descargar de Internet y se observan en un navegador. Los applets se descargan junto con una página HTML desde un servidor web y se ejecutan en la máquina cliente.
  - No tienen acceso a partes sensibles (por ejemplo: no pueden escribir archivos), a menos que uno mismo le dé los permisos necesarios en el sistema.
  - No tienen un método principal.

    - Son multiplataforma y pueden ejecutarse en cualquier navegador que soporte Java.

- **Servlets:**
    - Son componentes de la parte del servidor de Java EE, encargados de generar respuestas a las peticiones recibidas de los clientes.
    - Los servlets, al contrario de los applets, son programas que están pensados para trabajar en el lado del servidor y desarrollar aplicaciones Web que interactúen con los clientes.

- **Midlets:**
    - Son aplicaciones creadas en Java para su ejecución en sistemas de propósito simple o dispositivos móviles. Algunos juegos Java creados para teléfonos móviles son midlets.
    - Son programas creados para dispositivos embebidos (se dedican a una sola actividad), más específicamente para la máquina virtual Java Micro Edition (Java ME).
    - Generalmente son juegos y aplicaciones que se ejecutan en teléfonos móviles.

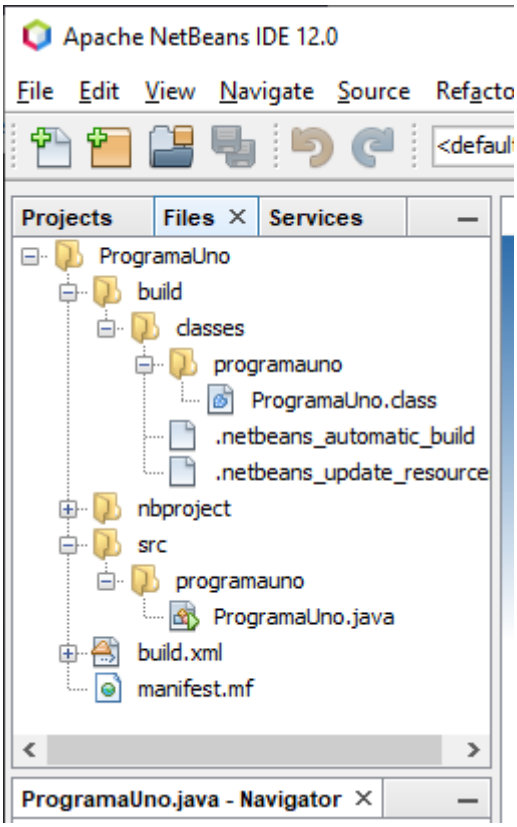
Un Applet es totalmente seguro, ya que no puede acceder, en ningún caso, a zonas sensibles del sistema. Es decir, no podría borrar o modificar nuestros archivos.

¿Verdadero o Falso?

¿Cuál de los siguientes entornos sólo está soportado en la plataforma Windows?

1. Eclipse.
2. IntelliJ IDEA.
3. Jcreator

Netbeans: Una de las ventajas que ofrece este entorno es poder examinar nuestros proyectos a través de la vista **Archivos**. Esta vista nos enseña la realidad de los archivos del proyecto, la carpeta **build** contiene los archivos compilados (.class), la carpeta **src** el código fuente y el resto, son archivos creados por Netbeans para comprobar la configuración del proyecto o los archivos necesarios para la correcta interpretación del código en otros sistemas (en cualquier caso, no hay que borrarlos). Para activar esta vista, selecciona en el menú principal Windows - Files.



José Javier Bermúdez Hernández. Uso educativo-nc. Elaboración propia.

Rellena los huecos con los conceptos adecuados:

En NetBeans, los archivos .class de un proyecto están alojados en la carpeta … y los .Java en la carpeta … .