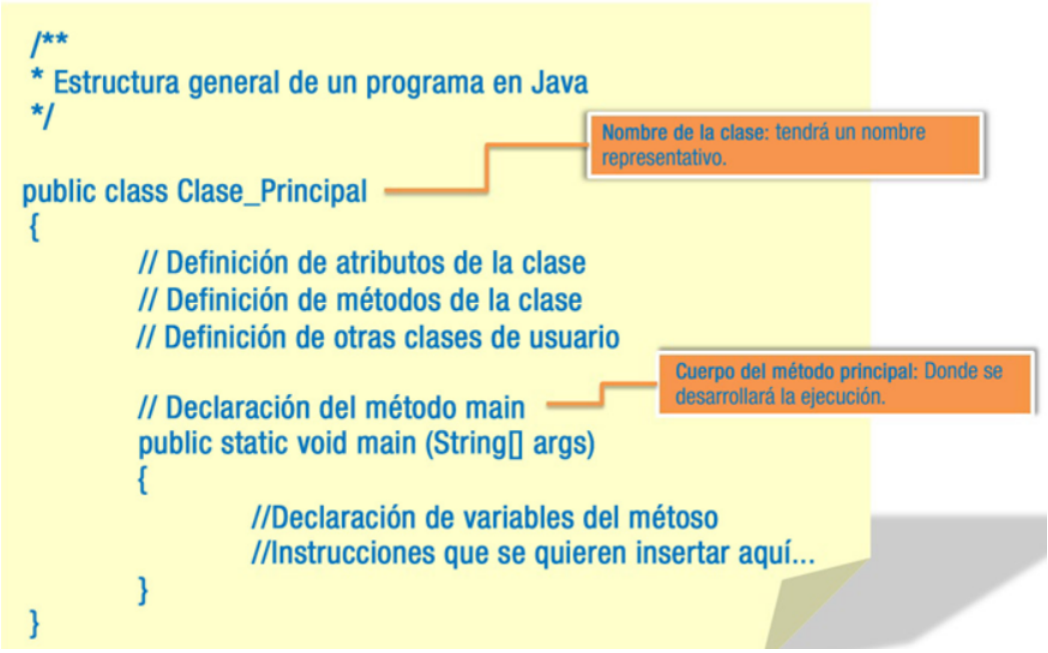
1. Programas en Java.

Hasta ahora, hemos descrito el lenguaje de programación Java, hemos hecho un recorrido por su historia y nos hemos instruido sobre su filosofía de trabajo.

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.



Vamos a analizar cada uno de los elementos que aparecen en dicho gráfico:

`public class Clase_Principal:` Todos los programas han de incluir una clase como esta. Es una clase general en la que se incluyen todos los demás elementos del programa. Entre otras cosas, contiene el método o función `main()` que representa al programa principal, desde el que se llevará a cabo la ejecución del programa. Esta clase puede contener a su vez otras clases del usuario, pero sólo una puede ser `public`. El nombre del fichero `.Java` que contiene el código fuente de nuestro programa, coincidirá con el nombre de la clase que estamos describiendo en estas líneas.

Recomendación

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.

· `public static void main (String[] args)`: Es el método que representa al programa principal, en él se podrán incluir las instrucciones que estimemos oportunas para la ejecución del programa. Desde él se podrá hacer uso del resto de clases creadas. Todos los programas Java tienen un método `main`.

· **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** `//` y **con** `/* */`. Con la primera forma estaríamos estableciendo una única línea completa de comentario y, con la segunda, 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 no hemos incluido ninguna línea de código que termine con punto y coma, 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.

1.2. El entorno básico de desarrollo Java.

Ya conoces cómo es la estructura de un programa en Java, pero, ¿qué necesitamos para llevarlo a la práctica? La herramienta básica para empezar a desarrollar aplicaciones en Java es el **JDK (Java Development Kit** o *Kit de Desarrollo Java*), que incluye un *compilador* y un *intérprete*

para línea de comandos. Estos dos programas son los empleados en la precompilación e interpretación del código.

Como veremos, existen diferentes entornos para la creación de programas en Java que incluyen multitud de herramientas, pero por ahora nos centraremos en el entorno más básico, extendido y gratuito, el **Java Development Kit (JDK)**. Según se indica en la propia página web de Oracle, JDK es un entorno de desarrollo para construir aplicaciones, applets y componentes utilizando el lenguaje de programación Java. Incluye herramientas útiles para el desarrollo y prueba de programas escritos en Java y ejecutados en la Plataforma Java.

Así mismo, junto a JDK se incluye una implementación del *entorno de ejecución* Java, el **JRE (Java Runtime Environment)** para ser utilizado por el JDK. El JRE incluye la Máquina Virtual de Java (MVJ ó JVM – Java Virtual Machine), bibliotecas de clases y otros ficheros que soportan la ejecución de programas escritos en el lenguaje de programación Java.

Para poder utilizar JDK y JRE es necesario realizar la descarga e instalación de éstos. Para ello podemos ir a la página (https://www.oracle.com/technetwork/java/javase/downloads/index.html)

Tras la instalación, los ficheros propios de ésta, se encontrarán en la ruta: "**C:\Program Files\Java\jdk-10.0.2**", y concretamente los ficheros ejecutables con los que interactuaremos se encuentran en la carpeta "**bin**". Serán, para ser más precisos, **javac.exe** y **java.exe

Para poder desarrollar nuestros primeros programas en Java sólo necesitaremos un editor de texto plano y los elementos que acabamos de instalar a través de Java SE. Pero para ello debemos tener la variable de **entorno PATH** debidamente configurada. Esto se trata en el apartado "Afinando la configuración". Gracias a ello, podremos ejecutar desde cualquier ruta, en la consola, las aplicaciones necesarias para compilar y generar el fichero **.class** y correr este ejecutable generado.

1.3. La API de Java.

Junto con el kit de desarrollo que hemos descargado e instalado anteriormente, vienen incluidas gratuitamente todas las bibliotecas de la API (Aplication 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 APIs 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.

Para saber más

Si quieres acceder a la información oficial sobre la API de Java, te proponemos el siguiente enlace (está en Inglés).

Información oficial sobre la API de Java

1.4. Afinando la configuración.

Para que podamos compilar y ejecutar ficheros Java es necesario que realicemos unos pequeños ajustes en la configuración del sistema. Vamos a indicarle dónde encontrar los ficheros necesarios para realizar las labores de compilación y ejecución, en este caso `Javac.exe` y `Java.exe`, así como las librerías contenidas en la API de Java y las clases del usuario.

La variable PATH: como aún no disponemos de un IDE (Integrated Development Environment - Entono Integrado de Desarrollo) la única forma de ejecutar programas es a través de línea de comandos. Pero sólo podremos ejecutar programas directamente si la ruta hacia ellos está indicada en la variable PATH del ordenador. Es necesario que incluyamos la ruta hacia estos programas en nuestra variable PATH. Esta ruta será el lugar donde se instaló el JDK hasta su directorio `**bin**`.

En Windows 10:

Vamos a las propiedades del sistema:

1. Presiona la tecla de Windows + X
2. Selecciona la opción «Sistema» y Propiedades avanzadas del sistema
3. Aparecerá una ventana con las propiedades del sistema

1.5. Codificación, compilación y ejecución de aplicaciones.

Una vez que la configuración del entorno Java está completada y tenemos el código fuente de nuestro programa escrito en un archivo con extensión `.java`, la compilación de aplicaciones se realiza mediante el programa **javac** incluido en el software de desarrollo de Java.

Para llevar a cabo la compilación desde la línea de comandos, escribiremos:

`javac archivo.java`

Donde `javac` es el compilador de Java y `archivo.java` es nuestro código fuente.

El resultado de la compilación será un archivo con el mismo nombre que el archivo Java pero con la **extensión** `**class**`. Esto ya es el archivo con el código en forma de *bytecode*. Es decir con el código precompilado. Si en el código fuente de nuestro programa figuraran más de una clase, veremos como al realizar la compilación se generarán tantos archivos con extensión `.class` como clases tengamos. Además, si estas clases tenían método `main` podremos ejecutar dichos archivos por separado para ver el funcionamiento de dichas clases.

Para que el programa pueda ser ejecutado, siempre y cuando esté incluido en su interior el método main, podremos utilizar el interprete incluido en el kit de desarrollo.

La ejecución de nuestro programa desde la línea de comandos podremos hacerla escribiendo:

`java archivo.class`

Donde `Java` es el intérprete y `archivo.class` es el archivo con el código precompilado.

Ejercicio resuelto

Vamos a llevar a la práctica todo lo que hemos estado detallando a través de la creación, compilación y ejecución de un programa sencillo escrito en Java.

Observa el código que se muestra más abajo, seguro que podrás entender parte de él. Cópialo en un editor de texto, respetando las mayúsculas y las minúsculas. Puedes guardar el archivo con extensión `.java` en la ubicación que prefieras. Recuerda que el nombre de la clase principal (en el código de ejemplo `MiModulo`) debe ser exactamente igual al del archivo con extensión `.java`, si tienes esto en cuenta la aplicación podrá ser compilada correctamente y ejecutada.

```
/* La clase MiModulo implementa una aplicación que simplemente imprime *   "Módulo profesional - Programación" en
pantalla.*/

class MiModulo {
        public static void main(String[] args) {
                // Muestra la cadena de caracteres.
                System.out.println("Módulo profesional - Programación");
        }
}
```

- Creamos el fichero MiModulo.java en una ubicación cualquiera, por ejemplo en una carpeta "
- EjerciciosJava" dentro de C:
- Podemos usar un editor de texto plano como el bloc de notas de Windows:
- Accede a la línea de comandos y teclea, en la carpeta donde has guardado el archivo Java, el comando **para compilarlo**: `Javac MiModulo.java`
- Aparentemente no hace nada, ya que no devuelve ningún mensaje. Precisamente esto es síntoma de que todo ha ido bien.

El compilador habrá generado entonces un fichero de código de bytes: `MiModulo.class`. Si visualizas ahora el contenido de la carpeta verás que en ella está el archivo `.java` y uno o varios (depende de las clases que contenga el archivo con el código fuente) archivos `.class`.

- Finalmente, **para realizar la ejecución** del programa debes utilizar la siguiente sentencia: `java MiModulo.java`

Si todo ha ido bien, verás escrito en pantalla: `"Módulo profesional - Programación"`.

1.6. Tipos de aplicaciones en Java.

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**. Los 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 **MicroEdition (Java ME)**.
- Generalmente *son juegos y aplicaciones que se ejecutan en teléfonos móviles*.