1. Introducción a la programación

Autor	Ⓧ Xerach Casanova
Clase	Programación
Fecha	@Dec 7, 2020 2:45 PM

1. Introducción

Las acciones de la vida cotidiana están relacionadas con la programación y el tratamiento de la información, de una u otra manera. El volumen de datos que se maneja constituye un vastato territorio en el que los programadores tienen mucho que decir.

2. Programas y programación

2.1. Buscando una solución

Como en la vida real, la programación se basa en la búsqueda y obtención de soluciones a un problema determinado.

- **Análisis del problema.** Debe ser definido y comprendido claramente para anlizarlo con detalle.
- **Diseño o desarrollo de algoritmos:** Se busca una solución sin entrar en detalles tecnológicos.
- **Resolución del algoritmo elegido en la computadora:** Convertimos el algoritmo en programa y al ejecutarlo se comprueba si se soluciona.

La solución al problema debe tener las siguientes virtures:

- **Corrección y eficacia:** si se resuelve el problema adecuadamente.

- **Eficiencia:** si lo hace en tiempo mínimo y con uso óptimo de recursos.

Para llegar a la solución debemos tener en cuenta los siguientes conceptos.

- **Abstracción.** Se analiza el problema y se descompone en problemas más pequeños y de menor complejidad que se describen de manera precisa.
- **Encapsulación:** Se oculta la información de los diferentes elementos que forman el sistema.
- **Modularidad:** Se divide el proyecto en módulos independientes que tendrán su función correspondiente.

2.2. Algoritmos y programas

Un algoritmo es una secuencia ordenada de pasos descrita sin ambigüedades que conducen a la solución de un problema.

Son independientes del lenguaje de programación. Un algoritmo llevado a distintos lenguajes debe dar siempre la misma solución.

Los lenguajes de programación son el medio para expresar un algoritmo. El diseño de los algoritmos parte de la creatividad del desarrollador y de los conocimientos de las técnicas de programación. Un mismo problema puede tener algoritmos distintos igualmente válidos.

Los algoritmos deben cumplir las siguientes características:

- **Ser precisos** e indicar el orden paso a paso.
- **Estar definido.** Al ejecutarlo más de una vez con los mismos datos de entrada se debe obtener el mismo resultado. También debe dar respuesta a cualquier dato de entrada.
- **Ser finito.** Debe terminar.

Técnicas para representar gráficamente algoritmos:

- **Diagramas de flujo.** Símbolos gráficos para su representación. Se utiliza en fase de análisis).
- **Pseudocódigo.** Técnica basada en el uso de palabras clave en lenguaje natural, constantes, variables, otros objetos, instrucciones y estructuras de programación para expresar la silución del problema.
- **Tablas de decisión.** Técnica de apoyo al pseudocódigo en situaciones compleja que se basa en una tabla se representan las posibles condiciones

del problema con sus respectivas acciones.

3. Paradigmas de la programación

Es un modelo básico para el diseño y la implementación de programas. Este modelo determina como será el proceso de diseño y la estructura final el programa.

- **Programación declarativa.** Consiste en decirle a un programa lo que tiene que hacer en lugar de decirle como hacerlo. SQL está basado en este paradigma.
- **Programación imperativa.** Se basa en desarrollar algoritmos detallando de forma clara y específica los comandos para llegar a la solución. Se basa en variables, tipos de datos, expresiones y estructuras de control de flujo. Dentro de ella encontramos la programación convencional, estructurada, orientada a objetos, orientada a eventos, orientada a aspectos...

Los lenguajes pueden soportar múltiples paradigmas o estar solo basados en un paradigma.

4. Fases de la programación

El proceso de creación de software se divide en tres fases:

- Fase de resolución del problema
- Fase de implementación
- Fase de explotación y mantenimiento.

4.1. Resolución del problema

El problema debe ser difinido y comprendido para poder analizarse. La fase de resolución pasa por dos etapas.

Análisis

En esta fase se dan por conocidas todas las necesidades que precisa la aplicación. Se especifican los procesos, estructuras y datos a emplear.

Este análisis ofrecerá una idea general de lo que se solicita y posteriormente se refinará para dar respuesta a las siguientes cuestiones.

- Cuál es la información que ofrecerá la resolución del problema - Salida de datos.
- Qué datos son necesarios para resolverlo - Entrada de datos.

En esta fase debemos analizar la documentación de la empresa, investigar, observar todo lo que rodea y recompilar información útil.

Diseño

La fase de análisis se convierte en un diseño más detallado, se indica la secuencia lógica de instrucciones para resolver el problema. En resumen, se construyen algoritmos.

En esta fase la aplicación se plantea como una operación global y se descompone en operaciones sencillas, detalladas y específicas. Estas operaciones se asignan a módulos separados.

Antes de pasar a la siguiente fase debemos realizar prueba o traza del programa utilizando datos concretos. Si tiene errores debemos volver a la fase de análisis.

4.2 Implementación

Es la fase donde se lleva a la realidad nuestro algoritmo. Esto implica varias etapas, que son:

Codificación o construcción

Traducimos los algoritmos a un determinado lenguaje de programación. Para comprobar la calidad y estabilidad se realizan pruebas que comprueban las funciones de cada módulo (pruebas unitarias), que los módulos funcionan entre ellos (pruebas de interconexión) y que todos funcionan bien en conjunto (pruebas de integración).

Seguidamente se compila. La compilación es el proceso en el que se traducen las instrucciones escritas en lenguaje de programación en lenguaje máquia (código binario). Y se lleva a cabo de dos formas:

- **A través de un compilador.** Programa que traduce completamente el código realizando un análisis lexicográfico, semántico y sintáctico, genera un código intermedio y finalmente genera el código máquina ejecutable.
- **A través de un intérprete**. Programa capaz de analizar y ejecutar programas escritos en lenguajes de alto nivel. Los intérpretes realizan la

traducción del programa a medida que sea necesaria. No se guarda el resultado de la traducción.

Una vez traducido el programa puede ser ejecutado.

Prueba de ejecución y validación

Se implanta la aplicación en el sistema donde va a funcionar, se pone en marcha y se comprueba su funcionamiento. Se analiza si responde a los requerimientos especificados, si se detectan nuevos errores y si la interfaz es amigable.

No se puede vanzar mientras se detectan errores. El testeado se documenta mediante:

- **Documentación interna:** encabezados, descripciones, delcaraciones del problema y comentarios que se incluyen en el código fuente.
- **Documentación externa:** manuales del programa.

4.3. Explotación

La fase de explotación es aquella en la que ya está siendo de utilidad para los usuarios. En ella se realizan evaluaciones periódicas, se llevan a cabo modificaciones e incluso se corrigen errores no detectados anteriormente. Para ello debemos tener a mano una documentación adecuada que facilite la comprensión y el uso del programa.

5. Ciclo de vida del software

Es la sucesión de estados o fases por las cuales pasa un software a lo largo de su vida:

- Especificación y análisis de requisitos.
- Diseño.
- Codificación.
- Compilación.
- Pruebas.
- Instalación y expoltación.
- Mantenimiento.

6. Lenguajes de programación.

Un lenguaje de programación es un conjunto de reglas sintácticas, semánticas, símbolos y palabras especiales establecidas para la construcción de programas. Es un lenguaje artificial, una construcción mental del ser humano para expresar programas.

La gramática del lenguaje son reglas aplicables al conjunto de símbolos y palabras para la construcción de sentencias correctas. Dispone de:

- **Léxico:** Vocabulario del lenguaje.
- **Sintaxis:** Combinación de símbolos y palabras especiales.
- **Semántica:** es el significado de cada construcción del lengaje.

En función de lo cerca que esté el lenguaje de programación del lenguaje humano o del ordenador se clasifican en distintos niveles.

6.1. Lenguaje máquina

O código binario. Es el lenguaje utilizado directamente por el procesador y el que puede interpretar un circuito microprogramable.

Fue el primer lenguaje utilizado de programación pero presenta los siguientes inconvenientes:

- Cada programa era válido solo para un tipo de procesador u ordenador.
- Lectura e interpretación extremadamente difícil, modificaciones costosas.
- Los programadores de la época debían memorizar largas combinaciones de ceros y unos.
- Los programadores se encargaban de introducir los códigos binarios en el ordenador.

6.2 Lenguaje ensamblador

Es la evolución directa del lenguaje máquina. Las secuencias binarias se sustituyen por códigos de operación elemntales llamados mnemotécnicos.

También presenta múltiples dificultades:

- Los programas siguen dependiendo del hardware que los soporta.

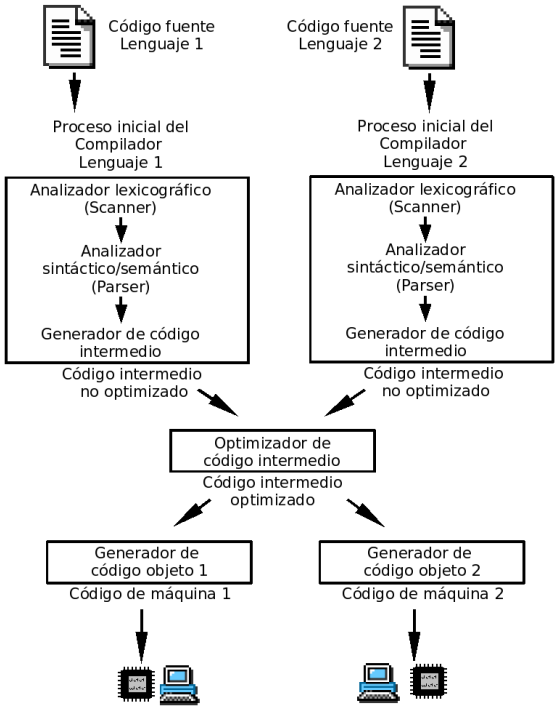
- Los programadores debían conocer detalladamente la máquina en la que programan para utilizar sus recursos.
- La lectura, interpretación y modificación seguía siendo difícil.

El lenguaje ensamblador necesita un intermediario que traduzca las instrucciones en lenguaje máquina.

6.3 Lenguajes compilados

Nacen para resolver los problemas derivados del uso del lenguaje ensamblador y acercar la programación al humano. Se denominan lenguajes de alto nivel y sus ventajas son:

- Fáciles de aprender.
- Reducción de tiempo y costes para desarrollar software.
- Independientes del hardware.
- La lectura, interpretación y modificación resulsa más sencilla.



Es menos eficiente que el código generado en lenguaje ensamblador y además debe traducirse en código máquina. Este trabajo lo hace el compilador.

El código objeto es el código máquina, el cual necesita añadir módulos de enlace o bibliotecas para obtener el código ejecutable.

6.4. Lenguajes interpretados

La ejecución se realiza a través de un intérprete. Cada instrucción se analiza, se traduce y se ejecuta al ser interpretada pero no se guarda en memoria.

El proceso de traducción y ejecución se llevan a cabo de manera simultánea. Presentan el inconveniente de ser algo más lentos y necesitan del programa intérprete ejecutándose.

A medio camino están los pseudo-compilados o pseudo-interpretados como Java, el cual se compila obteniendo el código binario en forma de bytecodes, estructuras parecidas al lenguaje máquina. Estos ficheros se encarga de interpretarlos la máquina virtual, la cual lo traducirá en lenguae máquina.

7. Java

7.1. ¿Qué y como es java?

Es un lenguaje sencillo de aprender.

Su gran virtud es que es válido para cualquier plataforma ya que el código será ejecutado en la máquina virtual de java JVM, el cual interpreta el código convirtiéndolo en código específico de la plataforma que lo soporta: Write once, run everywhere.

Características del lenguaje java:

- El código generado por el compilador es independiente de la arquitectura.
- Orientado a objetos con gran cantidad de bibliotecas definidas.
- Sintaxis similar a C y C++
- Es distribuido, preparado para TCP/IP.
- Robusto, realiza comprobaciones de código en tiempo de compilación y ejecución.
- Seguridad garantizada ya que java no accede a zonas delicadas de memoria o de sistema.

La programación orientada a objetos nace para resolver los problemas que daba la programación estructurada, aunque estuviese dividida en programación modular. Los problemas se dividen en objetos que tienen propiedades e interactúan con otros objetos. El programador se centra en cada objeto para programar sus elementos y funciones.

7.2. Breve historia

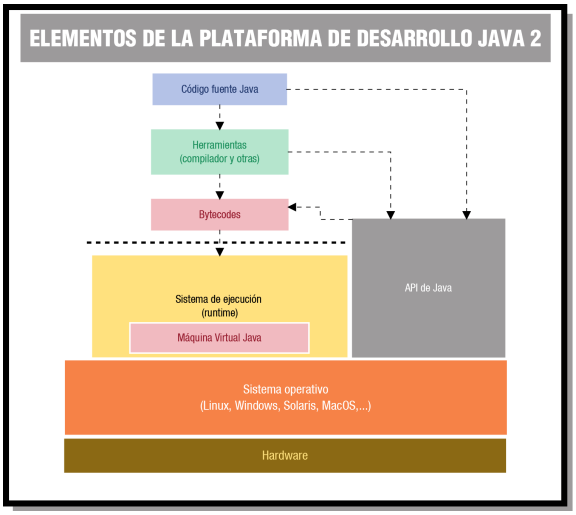
Java surgió en 1991 por un grupo de

ingenieros de Sun Microsystems para programar pequeños dispositivos electrónicos.

En 1995 pasa a llamarse Java. El factor determinante de su éxito fue la incorporación del intérprete java en la versión 2.0.

Para el desarrollo de programas es necesario utilizar el entorno JDK (Java Development Kit) el cual trae un compilador y un entorno de ejecución JRE (Java Run Enviroment) para los bytecodes generados.

Java 2 es la tercera versión del lenguaje e incluye:



- Lenguaje de programación Java
- Conjunto de bibliotecas (Java Core).
- Conjunto de herramientas para desarrollo de programas: compilador, generador de documentación, depurador...
- Entorno de ejecución o máquina virtual.

La plataforma Java 2 también incluyen otras funcionalidades:

- **J2SE:** relacionado con creación de aplicaciones y applets para ejecución en equipo y servidores
- **J2EE:** pensada para creación de aplicaciones web Java empresariales y del lado del servidor
- **Java FX:** crear e implementar aplicacionesd de internet enriquecidas.
- **Java Embedded:** creada para su ejecución en sistemas embebidas.
- **Java Card:** tecnología que permite ejecución de pequeñas aplicaciones java en tarjetas inteligentes.

7.3 La POO y Java

Los datos y el código (funciones o métodos) se combinan en entidades llamadas objetos. Estos tendrán un comportamiento (código) y estado (datos).

Los objetos permiten reutilización de código y son piezas reutilizables en proyectos distintos.

El paradigma de la programación orientada a objetos incluye:

- **Encapsulación.**
- **Herencia.**
- **Polimorfismo.**

Una **clase** es un prototipo o molde que indica que características van a tener los elementos creados a partir de ella. Una **instancia** es el objeto creado a partir de esta clase.

7.4. Independencia de la plataforma y trabajo en red

Una característica importante de java, además de la de independencia de la plataforma en que se ejecuta, es la posibilidad de crear aplicaciones que trabajan en red.

Esta capacidad ofrece múltiples posibilidades para la comunicación TCP/IP, con librerías que permiten acceso e interacción con HTTP, FTP, etc.

7.5. Seguridad y simplicidad.

Seguridad

Los accesos a zonas de memoria sensible como en C y C++ se han eliminado en java.

Además el código java es comprobado y verificado para que no se produzcan efectos no deseados.

Tampoco permite apertura de ficheros en máquina local ni ejecución de aplicación nativa.

Simplicidad

Es más potente que C o C++ pero más sencillo, eliminando la aritmética de punteros, registros, definición de tipos, gestión de memoria, etc.

Java borra automáticamente los objetos creados cuando determine que no se van a usar. A este proceso se le llama recolector de basura y permite al

programador liberarse de la gestión de memoria.

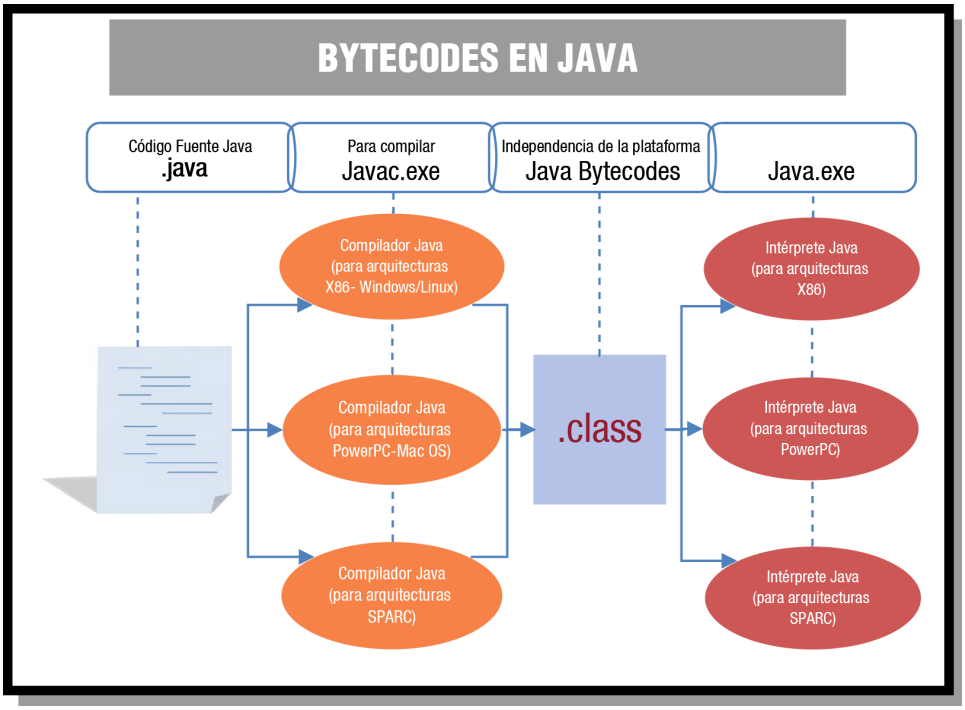
7.6 Java y los bytecodes

Una vez escrito el código, este se precompila generándose códigos de bytes (bytecodes) con extensión .class

Estos archivos son interpretados por la máquina virtual.

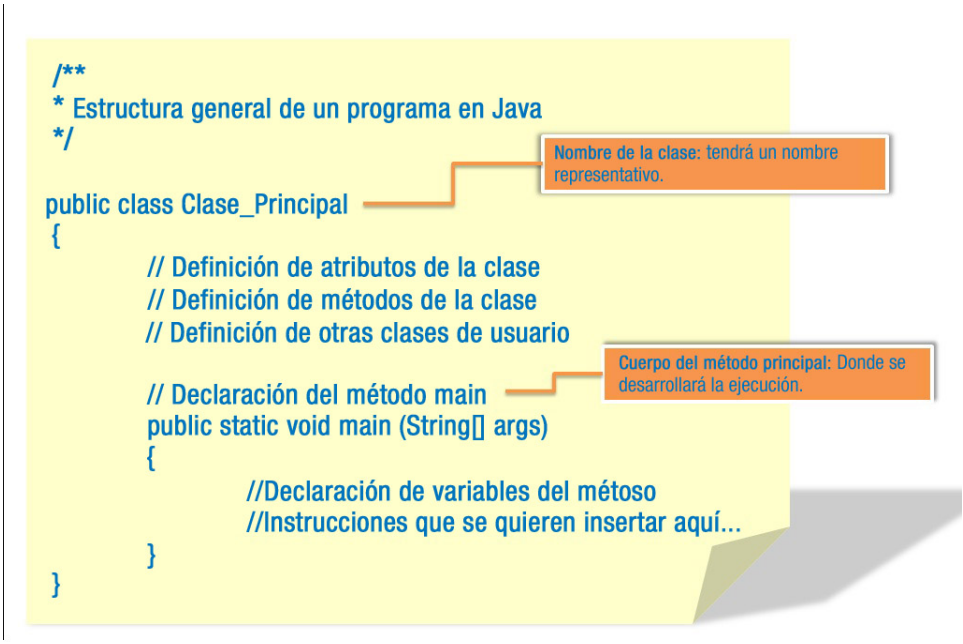
El proceso de precompilación consiste en verificar los códigos de bytes para asegurarse que:

- El código satisface las especificaciones de la máquina virtual.
- No existe amenaza contra el sistema.
- No se produce desbordamiento de memoria.
- Los parámetros y sus tipos son adecuados.
- No existen conversiones de datos no permitidas.



8. Programas java

8.1. Estructura de un programa.



- **public class Clase_Principal:** Es la clase general en la que se incluyen todos los elementos del programa, entre otras cosas contiene el método **main()**, que es el programa principal desde el que se llevará la ejecución del programa. **Solo puede ser public. El nombre del fichero .java coincide con el nombre de la clase.**
  - **public static void main (String[] args):** es el método que representa al programa principal, desde él se podrá usar el resto de clases. Todos los programas tienen un main.
  - **Comentarios:** se añaden entre /* */ para varias líneas o seguido de // para una sola línea.
  - **Bloques de código:** conjunto de instrucciones marcados entre llaves { }
  - **Punto y coma:** cada instrucción debe terminar en ; para no dar error.

8.2. El entorno básico de desarrollo Java

La herramienta básica para desarrollar aplicaciones en java es el JDK (Java Development Kit)). También incorpora el JRE (Java Runtime Environment) que incluye la máquina virtual (JVM) y la biblioteca de clases así como otros ficheros para la ejecución de programas.

8.3. La API de Java

Es la biblioteca de clases orientada a objetos gratuita de java. Proporciona paquetes de clases útiles para realizar múltiples tareas en un programa, se organiza en paquetes lógicos.

8.4. Tipo de aplicaciones Java

Aplicaciones de consola

- Programas independientes iguales que los creados con lenguajes tradicionales.
- Se componen como mínimo de un .class y un método main.
- No necesitan navegador web y se ejecutan invocando el comando java
- Las aplicaciones de consola se leen y escriben en la entrada y salida estándar, sin interfaz gráfica.

Aplicaciones gráficas

- Para incluir otros paquetes se utiliza la instrucción import. Por ejemplo para incluir todo el paquete swing se utiliza la instrucción import javax.swing.*;
- Utilizan clases con capacidades gráficas, como Swing (biblioteca gráfica)

Applets

- Programas incrustados en otras aplicaciones, normalmente en webs que muestra un navegador. Se descarga el applet y se ejecuta.
- Se descargan desde el servidor y se ejecutan desde la máquina del cliente.
- No acceden a partes sensibles a menos que se le de permisos necesarios en el sistema.
- No tienen método principal.
- Son multiplataforma.

Servlets

- Son componentes de la parte del servidor de Java EE, programas pensados para trabajar del lado de servidor y desarrollar webs que interactúen con el cliente.

Midlets

- Aplicaciones creadas para ejecución en sistemas de propósito simple o móviles, como los juegos java.

9. Entornos integrados de desarollor (IDE)

9.1. ¿Qué son?

Son aplicaciones para llevar a cabo el proceso completo de desarrollo de software a través de un único programa.

Podemos realizar labores de edición, compilación, depuración, detector de errores y ejecución. Además de otras capacidades únicas de cada entorno.

9.2. IDE's actuales

De libre distribución:

- Netbeans
- Eclipse
- BlueJ
- Jgrasp
- Jcreator LE

Propietarios o de pago:

- Intellij IDEA
- JBuilder
- JCreator
- JDeveloper

9.3. Netbeans

Es un producto libre y gratuito sin restricciones de uso. Proyecto de código abierto de gran éxito y comunidad numerosa.

Ofrece posibilidad de escribir en C, C++, Ruby, Groovy, Javascipt, CSS y PHP además de Jaava

Permite creaer J2EE

Permite crear aplicaciones Swing de forma sencilla

Permite crear aplicaciones JME para dispositivos móviles.

Mapa conceptual