Realizado por: Aitor Alguacil Maroto

Informe de Lenguajes, Paradigmas y Estándares de Programación

Índice:

1. Introducción

Los lenguajes, paradigmas y estándares en la programación son elementos cruciales para el desarrollo de software. La elección adecuada de un lenguaje y paradigma puede afectar la eficiencia y calidad del software, mientras que seguir estándares de programación significa una mejora en la legibilidad y la reducción de errores. Estos aspectos son fundamentales para la eficiencia, calidad, colaboración y mantenimiento en el desarrollo de software.

2. Tipos de Lenguajes de Programación

En la programación actualmente, se distinguen varios niveles según los lenguajes (alto, medio y bajo), estos niveles se basan sobre todo en el grado de abstracción y la proximidad al hardware que presenta un lenguaje respecto al sistema que ejecuta un programa. Los niveles sirven para distinguir y describir los tipos de relaciones que existen entre los lenguajes y el hardware que presenta un ordenador.

Tipos de lenguajes:

- Lenguajes de alto nivel:
  lenguajes de programación que ofrecen una abstracción significativa del hardware en el que se encuentran. Están diseñados para ser más fáciles de entender y escribir, permitiendo así a los programadores expresar sus ideas de manera más clara y concisa.
  Los lenguajes de alto nivel más utilizados actualmente son los siguientes:

  - Python: utilizado en desarrollo web, análisis de datos, inteligencia artificial y automatización.

  - Java: se utiliza ampliamente en el desarrollo de aplicaciones empresariales y Android.

  - JavaScript: utilizado en desarrollo web para crear interacciones dinámicas en los navegadores.

- Lenguajes de medio nivel:
  lenguajes de programación que presentan un nivel medio de abstracción y control, y funcionan mediante la traducción de un código escrito al lenguaje de máquina, mediante un proceso de compilación, para así poder ejecutarlo.
  Los lenguajes de medio nivel más usados a día de hoy son los siguientes:

- C: se utiliza para desarrollar sistemas operativos, controladores de hardware y aplicaciones de alto rendimiento.

- C++: extensión de C que agrega características de programación orientada a objetos y se utiliza en el desarrollo de juegos, software de sistemas y aplicaciones de alto rendimiento.

- Rust: se centra en la seguridad y el rendimiento y es utilizado en aplicaciones donde la fiabilidad es crítica.

- Lenguajes de bajo nivel:
  lenguajes de programación que más se acercan al hardware, por lo que proporcionan un control extremo sobre el sistema y sus recursos. Se utilizan principalmente en la programación de sistemas, desarrollo de controladores y tareas de bajo nivel.
  Los lenguajes de bajo nivel más usados actualmente son los siguientes:

  - Lenguaje de máquina: lenguaje de programación de nivel más bajo que consiste en códigos binarios y hexadecimales directamente ejecutables por la CPU.

  - Ensamblador (Assembly): representa instrucciones específicas de la arquitectura del procesador. Se utiliza para programar directamente el hardware y es esencial en el desarrollo de sistemas operativos.

  - Fortran: utilizado principalmente en programación científica e ingeniería y aplicaciones de cálculos numéricos y simulaciones de alto rendimiento.

3. Paradigmas de Programación

Paradigma: ejemplo o modelo de algo.

Tipos de paradigmas en la programación:

- Paradigma Imperativo:
  Características distintivas: se centra en "cómo" se realiza una tarea. Los programas se construyen como una serie de instrucciones que modifican el estado de las variables y se utilizan estructuras de control (ciclos y condicionales) para controlar el flujo de ejecución.
  Consiste en explicar detenidamente y de una forma muy clara cómo funciona el código.
  Ejemplo de lenguaje: C, Pascal, Fortran.

- Paradigma Declarativo:
  Características distintivas: el paradigma declarativo se centra en "qué" se quiere lograr, es decir, prioriza la claridad del resultado antes que la claridad del paso a paso. Se describen relaciones, consultas o estructuras de datos sin detallar los pasos específicos para alcanzar el resultado.
  Ejemplo de lenguaje: SQL, HTML.

- Paradigma Orientado a Objetos (OO):
  Características distintivas: se modelan conceptos del mundo real como "objetos" que tienen atributos y comportamientos. Este tipo de paradigma se constituye por piezas simples u objetos que al relacionarse entre sí forman diferentes componentes del sistema que estemos trabajando.
  Ejemplo de lenguaje: Java, C++, Python.

- Paradigma Funcional:
  Características distintivas: En este paradigma, el enfoque está en "qué" se hace, no en "cómo". Se basa en funciones matemáticas puras que no tienen efectos secundarios. La recursión y la inmutabilidad son comunes, y las funciones son “ciudadanos de primera clase".
  Ejemplo de lenguaje: Haskell, Lisp.

- Paradigma Lógico:
  Características distintivas: se basa en reglas lógicas y relaciones. Los programas se construyen mediante consultas y hechos lógicos, permitiendo la inferencia automática de soluciones a través de la resolución de consultas basada en reglas. Es especialmente útil en sistemas expertos y procesamiento de lenguaje natural.
  Ejemplo de lenguaje: Prolog.

- Paradigma de Programación Estructurada:
  Características distintivas:
  El paradigma de programación estructurada se centra en dividir programas en módulos o funciones independientes con propósitos específicos. Emplea estructuras de control como secuencias, selecciones y bucles para crear un código claro y organizado.
  Este paradigma de programación se orienta más en mejorar la calidad, el tiempo de desarrollo y la claridad del código
  Ejemplo de lenguaje: C, Pascal.

4. Estándares de Programación

Los estándares de programación, establecen pautas y reglas que los programadores/desarrolladores deben tener en cuenta a la hora de hacer un código, facilitando así la colaboración, comprensión y el mantenimiento del mismo.
Los estándares se deben seguir con el objetivo de garantizar la calidad, estructura, consistencia y mantenibilidad del código.

Ejemplos de estándares:

- PEP8 (Python): Documento que proporciona pautas y mejores prácticas sobre cómo escribir código en Python.
- ESlint (JavaScript): Su tarea consiste en examinar el código de nuestra aplicación, identificar posibles problemas mediante patrones y, en la medida de lo posible, abordarlos por sí mismo.
- Java Code Conventions: conjunto de directrices y recomendaciones establecidas por Oracle, ofrece pautas para que el código Java sea más claro y consistente, facilitando la colaboración y mejorando la calidad del software.

Beneficios de utilizar a estándares:

- Mantenibilidad: el código es más fácil de mantener y actualizar, ya que sigue un formato y estilo consistentes.
- Colaboración: facilita la colaboración entre desarrolladores, ya que todos siguen las mismas reglas.
- Legibilidad: el código es más legible, lo que reduce errores y facilita la comprensión.
- Calidad: los estándares promueven buenas prácticas, lo que desemboca en un código de mayor calidad.

Consecuencias de no utilizar estándares:

- Dificultad de mantenimiento: el código desorganizado y sin formato consistente dificulta la corrección de errores y las actualizaciones.
- Confusión: los desarrolladores pueden tener dificultades para entender el código de otros, lo que ralentiza el desarrollo.
- Errores: la falta de estándares puede dar lugar a errores y problemas de calidad en el software.

- Desperdicio de tiempo: la corrección y refactorización constante del código mal formateado consume tiempo y recursos.

5. Conclusión

Comprender varios lenguajes, paradigmas y estándares en desarrollo de software es muy importante ya que cada uno de éstos aborda los posibles problemas de una manera única. La elección correcta puede optimizar el rendimiento, mantenimiento, legibilidad y colaboración en proyectos. La adaptación a estándares garantiza coherencia y calidad. La adaptación a la evolución tecnológica y la creatividad en la resolución de problemas son esenciales y el conocimiento profundo sobre estas herramientas permite desarrollar software eficiente y sostenible, fundamental en un mundo digital en constante cambio.

Referencias/Bibliografía:

1. Chat GPT
2. https://platzi.com/blog/paradigmas-programacion/
3. https://www.grupoatrium.com/actualidad/los-8-principales-usos-de-python/
4. https://www.tokioschool.com/noticias/para-que-sirve-programacion-java/
5. https://realpython.com/python-pep8/#:~:text=PEP%208%2C%20sometimes%20spelled%20PEP8,and%20consistency%20of%20Python%20code
6. https://platzi.com/tutoriales/1099-fundamentos-javascript-2017/2181-como-usar-eslint-y-prettier-para-mejorar-tu-codigo-javascript/#:~:text=ESLint%20es%20un%20linter%20de,Tambien%20analiza%20nuestro%20c%C3%B3digo%20JavaScript