2024

Lenguajes, Paradigmas y Estándares de Programación

PARTE 3

JUN XU CHENG

MSMK | PARA: GABRIELA GARCÍA

ÍNDICE

1. Introducción

En el dinámico mundo del desarrollo de software, los lenguajes de programación son los hilos que dan forma a la lógica y funcionalidad de las aplicaciones. Este informe se sumerge en el vasto universo de los lenguajes de programación, delineando su diversidad a través de diferentes niveles y categorías. Cada lenguaje desempeña un papel único en la creación de soluciones tecnológicas. Además, exploramos los paradigmas de programación, cada uno como un marco conceptual que guía la construcción de software, junto con la importancia de los estándares de programación para la coherencia y eficiencia del código.

2.Tipos de Lenguaje de Programación

Los lenguajes de programación son herramientas fundamentales para el desarrollo del software, ya que replican la lógica de los lenguajes humanos. Estos lenguajes se categorizan en diversos niveles, cada uno con características particulares.

2.1. Nivel Alto

Aspiran a la universalidad y pueden aplicarse en diversos tipos de sistemas. Pueden ser de propósito general o especifico, brindando flexibilidad a los programadores. Ejemplos:

- **Python**: Versátil, se utiliza en desarrollo web, inteligencia artificial y análisis de datos.
- **Java:** Utilizado en aplicaciones empresariales, desarrollo Android y sistemas embebidos.

2.2. Nivel Medio

Se sitúan entre los niveles bajo y alto, permitiendo operaciones de alto nivel y la gestión local de la arquitectura del sistema. Ejemplo:

- **C**: Ampliamente utilizado en sistemas operativos y desarrollo de software de bajo nivel.

Además de la clasificación por nivel, también encontramos categorías según la forma de programar:

- **Lenguajes Imperativos:** Menos flexibles, programan mediante órdenes condicionales y bloques de comandos. Ejemplo: **C++**
- **Lenguajes Funcionales:** Utilizan funciones que se requieren según la entrada recibida y son resultado de otras funciones. Ejemplo: **Haskell**

2.3. Nivel Bajo

Diseñados para hardware específico, no son transferibles a otros sistemas. Aunque son menos legibles y carecen de portabilidad. Ejemplo:

- **Ensamblador:** Utilizado para programación de bajo nivel, especialmente en sistemas embebidos.

2.4. Lenguajes según su propósito

En la programación hay dos tipos de lenguajes: los de propósito general como JavaScript, C++ o Python, que sirven para desarrollar más de un tipo de software; y los de propósito específico, como SQL, que se caracteriza por cumplir con tareas muy específicas.

2.5. Lenguajes según su método de ejecución

Existen dos tipos en base a su método de ejecución: los lenguajes compilados, como C, donde necesitaremos recompilar el programa para cada máquina en la que se ejecute[1], y los interpretados, como JavaScript, que leen y ejecutan el código directamente.

2.6. Lenguajes Web

El internet que conocemos se compone de servidores y dispositivos clientes. Para desarrollar software en ambos externos, se utilizan lenguajes de FrontEnd y BackEnd.

---

[1] Notion. (2024). *Notion – The all-in-one workspace for your notes, tasks, wikis, and databases.* [online] Available at: https://speckle-porpoise-32d.notion.site/Introducci-n-a-la-programaci-n-906cf521f30f41d9893cd2098b41cd87 [Accessed 24 Jan. 2024].

2.6.1. FrontEnd

El FrontEnd sirve para realizar la interfaz de un sitio web, la parte visible e interactiva. Entre los lenguajes más populares están: HTML, CSS, JavaScript, TypeScript, Elm o Swift.

2.6.2. BackEnd

El backend es el encargado de procesar toda la información que alimenta a un frontend.[2] Constituye la parte lógica interna de un sitio web, asegurando el funcionamiento correcto sin elementos visuales para el usuario.

3. Paradigmas de Programación

Un paradigma de programación es una manera o estilo de programación de software. Tratándose de un conjunto de métodos sistemáticos aplicables a todos los niveles del diseño de programas.

3.1. Paradigma Imperativo

Desarrolla programas mediante procedimientos y una serie de instrucciones paso a paso. Se centra en describir “cómo” hacer algo. Ejemplo:

- **C++:** Desarrollo de sistemas embebidos, videojuegos y software de sistemas operativos.

3.2. Paradigma Declarativo

Se centra en el resultado final desde el inicio, automatiza la determinación de la ruta para obtener una solución, solo requiere claridad en el proceso a seguir. Ejemplo:

- **SQL:** Manipulación y consulta de base de datos, análisis de datos y desarrollo web.

---

[2] Coppola, M. (2022). *Frontend y backend: qué son, en qué se diferencian y ejemplos*. [online] Hubspot.es. Available at: https://blog.hubspot.es/website/frontend-y-backend [Accessed 24 Jan. 2024].

3.3. Paradigma Orientado a Objetos

Ofrece modelo de objetos que representan elementos del problema a resolver, con características y funciones. Permite separar componentes, simplificar la creación, depuración y mejoras del programa. Ejemplo:

- **Java:** Desarrollo de aplicaciones empresariales, sistemas web y aplicaciones móviles.

3.4. Paradigma Funcional

Trabaja a través de funciones matemáticas, su uso más común es en el ámbito académico. Se enfoca en “que” se logra en lugar de “como”. Ejemplo:

- **Haskell:** Programación académica, procesamiento de datos complejos y sistemas de inteligencia artificial.

2.5. Paradigma Lógico

Un software programado según este principio contiene un conjunto de principios que se pueden entender como una recopilación de hechos y suposiciones. Ejemplo:

- **Prolog:** Uso en el ámbito de la inteligencia artificial.

4. Estándares de Programación

Los estándares de programación son reglas para crear un código limpio y eficiente. Facilitando la lectura y mantenimiento del código. Ejemplos:

- **Lenguaje - PHP:** Aplican las PSR, que son elegidas mediante votaciones.
    - **PSR -** 4: Establece un estándar para la autoloading de clases.
    - **PSR –** 2: Define el estándar del estilo de código.
- **Lenguaje - JavaScript:** ESLint busca patrones problemáticos y suele usarse para la integración con editores y la configurabilidad.
- **Lenguaje - Python:** PEP 8 establece directrices para el estilo del código, cubre temas como la indentación, comentarios y muchos más.

4.1. Beneficios de adherirse a Estándares

Cumplir con estándares ofrece beneficios como la legibilidad del código y facilidad de mantenimiento, además de mejorar el flujo de trabajo, la eficiencia y productividad incrementa y la calidad del producto mejora.

4.2. Consecuencia de No Hacerlo

No hacerlo puede crear un código confuso, propenso a errores dando preocupaciones de seguridad y dificulta la colaboración de proyectos.

5. Conclusión

En resumen, este análisis de lenguajes de programación, paradigmas y estándares destaca la diversidad esencial para el desarrollo de software. Desde la versatilidad de Python hasta la precisión de C, cada lenguaje y paradigma aporta una perspectiva única. Adherirse a estándares, como en PHP, JavaScript y Python, no solo mejora la legibilidad, sino que también impulsa la eficiencia y productividad. Ignorar estos estándares puede resultar en código propenso a errores y dificultades de colaboración. En este vasto paisaje, la elección informada y la adhesión a estándares son fundamentales para un desarrollo de software efectivo y sostenible.

6. Bibliografías

Griselda (2019). *Lenguajes de programación: el conocimiento base para dominar la informática*. [online] Escuela Fintech. Available at: https://escuelafintech.com/lenguajes-de-programacion/ [Accessed 3 Jan. 2024].

Isabel Cristina Reyes (2022). *5 Tipos de Paradigmas de Programación | CognosOnline Colombia*. [online] CognosOnline. Available at: https://cognosonline.com/co/blog/que-son-paradigmas-de-programacion/ [Accessed 3 Jan. 2024].

Miriam Martínez Canelo (2020). *¿Qué son los paradigmas de programación?* [online] Profile Software Services. Available at: https://profile.es/blog/que-son-los-paradigmas-de-programacion/ [Accessed 3 Jan. 2024].

Merino, M. (2020). *El lenguaje Prolog: un ejemplo del paradigma de programación lógica*. [online] Genbeta.com. Available at: https://www.genbeta.com/desarrollo/lenguaje-prolog-ejemplo-paradigma-programacion-logica [Accessed 3 Jan. 2024].

Rupali López Echeverría (2021). *La importancia de usar buenas prácticas de programación*. [online] Medium. Available at: https://medium.com/@pisc1s/la-importancia-de-usar-buenas-pr%C3%A1cticas-de-programaci%C3%B3n-6bbf06833ab2 [Accessed 3 Jan. 2024].

losapuntesde (2023). *Estándares de estilo y buenas prácticas en Python: Uniformidad y calidad de código*. [online] Apuntes de Programador. Available at: https://apuntes.de/python/estandares-de-estilo-y-buenas-practicas-en-python-uniformidad-y-calidad-de-codigo/#gsc.tab=0 [Accessed 3 Jan. 2024].

ESLint - Pluggable JavaScript linter. (2015). *List of available rules*. [online] Available at: https://denar90.github.io/eslint.github.io/docs/rules/ [Accessed 3 Jan. 2024].

Github.io. (2024). *PHP - Estandares*. [online] Available at: https://coppeldev.github.io/php/standards/ [Accessed 3 Jan. 2024].

Notion. (2024). *Notion – The all-in-one workspace for your notes, tasks, wikis, and databases.* [online] Available at: https://speckle-porpoise-32d.notion.site/Introducci-n-a-la-programaci-n-906cf521f30f41d9893cd2098b41cd87 [Accessed 24 Jan. 2024].