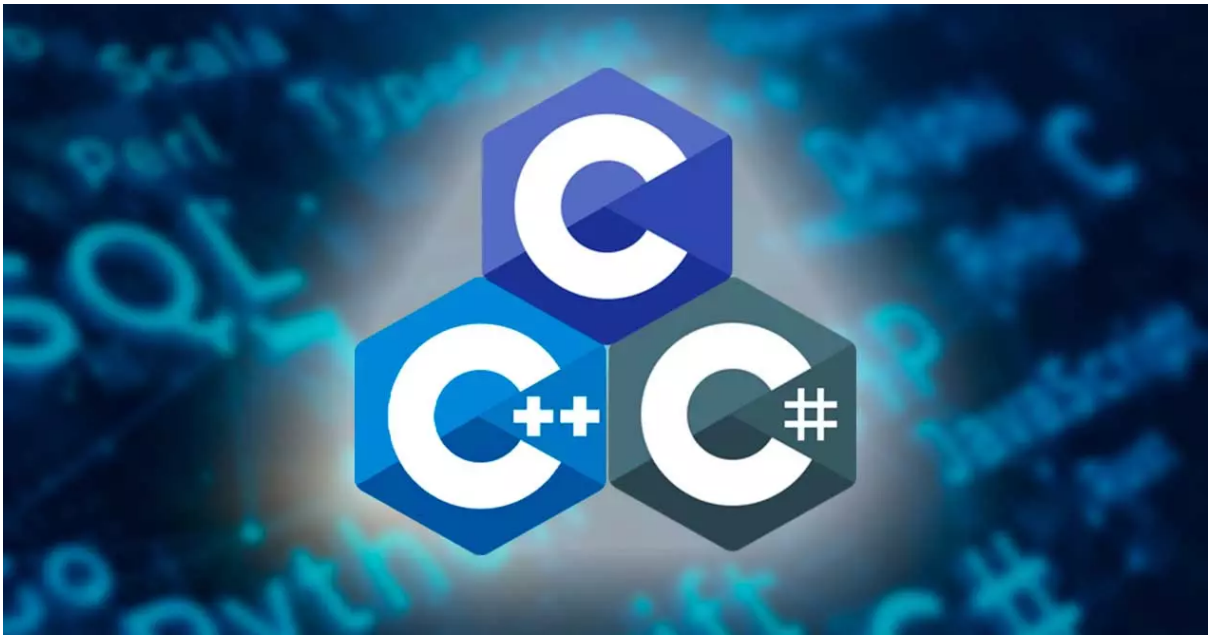
VideoTarea - C, C++ y C#



Iker Teja Canales

17/12/2023

2º DAM, Acceso a datos

Índice

Evolución de C.....3

Evolución de C++..... 4

Evolución de C#.....5

Principales características de C..... 6

Principales características de C++..... 7

Principales características de C#..... 8

Sintaxis C, C++ y C#..... 9

Aportaciones personales y conclusión.....11

Referencias.....12

Evolución de C

Creación en los Años 70:

En la década de 1970, Dennis Ritchie creó C en Bell Labs con la intención de desarrollar UNIX, enfocándose en la simplicidad y eficiencia.

Estándar ANSI C (1989):

En 1989, se estableció el estándar ANSI C, formalizando características y mejorando la portabilidad, brindando una base común para su uso.

C99 (1999):

La versión C99 (1999) introdujo características como variables de longitud variable y comentarios en estilo de C++, mejorando la flexibilidad del código.

Rol Fundamental en UNIX:

Desde sus inicios, C desempeñó un papel crucial en el desarrollo de UNIX, adaptándose a diversas arquitecturas y contribuyendo al éxito del sistema operativo.

Mejoras Continuas:

A lo largo de los años, C ha experimentado mejoras centradas en optimizaciones de código y gestión de punteros, manteniendo su relevancia en el desarrollo de software.

Expansión en Sistemas Embebidos:

La simplicidad y eficiencia de C lo han convertido en un pilar en sistemas embebidos y de bajo nivel, adaptándose a entornos con recursos limitados.

Hoy en día, C sigue siendo esencial en sistemas operativos, controladores de hardware y programas de bajo nivel, destacando por su evolución constante y su adaptabilidad a las necesidades cambiantes.

Evolución de C++

Orígenes y Extensión de C (1980):

En la década de 1980, Bjarne Stroustrup creó C++ como una extensión de C, incorporando programación orientada a objetos sin perder eficiencia.

Estándar C++98 (1998):

Establecido en 1998, C++98 formalizó características clave, marcando la transición completa a un lenguaje orientado a objetos.

C++11, C++14, C++17:

Estándares posteriores, como C++11, C++14 y C++17, introdujeron mejoras significativas, como lambdas y expresiones regulares.

Programación Genérica y STL:

C++ se enriqueció con programación genérica y la Biblioteca Estándar (STL), facilitando código versátil con estructuras de datos y algoritmos listos para usar.

Diversificación de Aplicaciones:

Utilizado en sistemas embebidos, desarrollo de juegos y proyectos complejos, C++ demostró su versatilidad.

Influencia Recíproca con C#:

C++ y C# se han influenciado mutuamente, compartiendo conceptos y enfoques en su evolución.

Continua Innovación:

La evolución de C++ continúa con estándares nuevos y propuestas para mejorar rendimiento, seguridad y funcionalidades, manteniendo su relevancia en el desarrollo de software.

Evolución de C#

Inicios en la Década de 2000:

Concebido por Microsoft en los primeros años de la década de 2000 como respuesta a la necesidad de un lenguaje moderno y orientado a objetos para la plataforma .NET.

Estandarización (2001):

En 2001, C# obtuvo estatus de estándar por parte de ECMA, allanando el camino para su adopción más allá del entorno Microsoft.

Versiones Sucesivas:

A lo largo de los años, C# ha experimentado iteraciones con mejoras y nuevas características para hacer más eficiente y expresivo el desarrollo de software.

Orientación a Objetos y Funcionalidades (2007):

Desde la versión 3.0 (2007), C# incorporó elementos de programación funcional, ampliando su versatilidad y capacidad de expresión en el código.

Integración con el Ecosistema Microsoft:

C# se integró estrechamente con herramientas y tecnologías de Microsoft, convirtiéndose en el lenguaje preferido para el desarrollo en la plataforma Windows y aplicaciones empresariales.

Interoperabilidad y .NET Core/.NET 5:

A partir de C# 7.0, se enfocó en interoperabilidad y evolución de .NET Core, consolidándose como .NET 5 en 2020 con mejoras y soporte multiplataforma.

La evolución de C# ha sido tanto técnica como estratégica, adaptándose a las demandas de la industria.

Principales características de C

CARACTERÍSTICA	DESCRIPCIÓN
Sencillo y Eficiente	Fácil de entender y escribir, con una sintaxis directa para expresar ideas claramente.
Control Directo de la Memoria	Proporciona a los programadores control preciso sobre la memoria, útil para tareas específicas.
Portabilidad	El código en C es versátil y puede utilizarse en diferentes sistemas, facilitando la creación de programas.
Rápido	Diseñado para ejecutar programas velozmente al traducirse directamente al lenguaje de la computadora.
Programación Procedimental	Organiza el código en funciones y procedimientos, brindando flexibilidad y claridad en la escritura.
Usado en Sistemas Operativos	Común en el desarrollo de sistemas operativos, como UNIX, gracias a su manejo directo del hardware.
Reutilización del Código	Facilita la creación de partes de código reutilizables para construir programas más grandes.

Principales características de C++

CARACTERÍSTICA	DESCRIPCIÓN
Capacidad de utilizar Bibliotecas	Son conjuntos de funciones predefinidas, para facilitar y acelerar la escritura de código. Los desarrolladores pueden aprovechar el trabajo ya realizado por otros usuarios .
Orientado a Objetos	Facilita la manipulación y configuración de sus parámetros o propiedades.
Rapidez	La compilación y ejecución de programas en C++ es más rápida en comparación con otros lenguajes.
Compilación	Requiere la compilación del código de bajo nivel antes de la ejecución, a diferencia de algunos lenguajes.
Punteros	Hereda la capacidad de usar punteros de C, proporcionando una herramienta poderosa para manipulaciones directas de memoria lo que puede ser útil en optimizaciones de rendimiento.
Didáctico	Aprender programación en C++ facilita la comprensión de otros lenguajes como Java, C#, PHP, Javascript, etc.

Principales características de C#

CARACTERÍSTICA	DESCRIPCIÓN
Plataforma .NET	C# se ejecuta en el entorno de tiempo de ejecución de .NET, permitiendo la portabilidad de los programas en diferentes plataformas.
Recogida de Basura	Utiliza un recolector automático de basura que gestiona eficientemente la memoria, liberando objetos no utilizados y simplificando la administración de recursos.
Fácil Integración con Windows	Está estrechamente integrado con las herramientas de desarrollo de Microsoft, facilitando la creación de aplicaciones para el sistema operativo Windows.
Modernidad	Incorpora características actuales, como tipos de datos avanzados, expresiones lambda y manejo de eventos, mejorando la eficiencia y expresividad del código.
Seguridad de Tipos	C# es un lenguaje fuertemente tipado, lo que significa que requiere la declaración y el respeto estricto de los tipos de datos, evitando errores difíciles de detectar.

Sintaxis C, C++ y C#

Este programa básico ilustra cómo se realiza la entrada de datos, la manipulación y la salida en cada uno de los lenguajes.

C:

```c
#include <stdio.h>

int main() {
    int numero;

    printf("Ingrese un número: ");
    scanf("%d", &numero);
    printf("El doble del número es: %d\n", numero * 2);

    return 0;
}
```

C++:

```cpp
#include <iostream>

int main() {
    int numero;

    std::cout << "Ingrese un número: ";
    std::cin >> numero;
     std::cout << "El doble del número es: " << numero * 2
<< std::endl;

    return 0;
}
```

C#:

```csharp
using System;

class Program {
    static void Main() {
        Console.Write("Ingrese un número: ");
        int numero = Convert.ToInt32(Console.ReadLine());
        Console.WriteLine("El doble del número es: " +
(numero * 2));
    }
}
```

Aportaciones personales y conclusión

Personalmente, encuentro que los lenguajes de programación C, C++, y C# ofrecen un conjunto único de características que se adapta a diferentes necesidades y contextos de desarrollo.

En el caso de C, su simplicidad y eficiencia continúan siendo fundamentales en el desarrollo de sistemas operativos y programas de bajo nivel. La capacidad de controlar la memoria es necesario en entornos donde la optimización es crucial.

C++, por su parte, ha ampliado significativamente las posibilidades de C al introducir la programación orientada a objetos y la Biblioteca Estándar de C++. La combinación de características ha posibilitado el desarrollo de sistemas complejos y la creación de aplicaciones diversificadas.

En cuanto a C#, su integración con la plataforma .NET y su enfoque en la programación orientada a objetos lo convierten en una opción poderosa para el desarrollo de aplicaciones empresariales en entornos Windows.

Conclusión final:

Cada lenguaje tiene su lugar y propósito específico, y la elección entre ellos dependerá de los requisitos del proyecto y las preferencias del desarrollador. La capacidad de estos lenguajes para adaptarse y seguir siendo relevantes a lo largo del tiempo subraya su importancia continua en el panorama de la programación.

Referencias

https://imaginaformacion.com/diccionario-informatico/c-c-y-c-diferencias

https://desarrolloweb.com/home/c

https://es.wikipedia.org/wiki/C_(lenguaje_de_programaci%C3%B3n)

https://openwebinars.net/blog/que-es-cpp/

https://openwebinars.net/blog/que-es-c/

https://es.wikipedia.org/wiki/C%2B%2B#:~:text=C%2B%2B%20es%20un%20lenguaje,C%2B%2B%20es%20un%20lenguaje%20h%C3%ADbrido.

https://www.tokioschool.com/noticias/c-que-es/

https://www.timetoast.com/timelines/evolucion-del-lenguaje-c-6e07f339-ba98-46df-b647-b91d53b37f36

https://learn.microsoft.com/es-es/dotnet/csharp/tour-of-csharp/

https://es.wikipedia.org/wiki/C_Sharp

https://es.slideshare.net/krisnaready/sintaxis-del-lenguaje-c