**Preguntas para una entrevista para el puesto de desarrollador Junior en C#:**

**P: ¿Qué es C#?**

Respuesta:

C# es un lenguaje orientado a objetos y con seguridad de tipos que utiliza un enfoque orientado a componentes para desarrollar aplicaciones .NET. El lenguaje es compilado por el marco .NET en Microsoft Intermediate Language, lo que lo convierte en un lenguaje de programación orientado a objetos de elección.

Con C#, puede crear varias aplicaciones, incluidas aplicaciones de Windows, servicios de API basados ​​en la nube, aplicaciones móviles para plataformas iOS y Android, software de IA y aprendizaje automático (ML), aplicaciones de blockchain y aplicaciones sin servidor.

**P: ¿Cuáles son algunas de las características clave de C#?**

Respuesta:

Simplicidad: C# no tiene punteros ni permite la modificación directa de la memoria, lo que lo distingue de lenguajes como C++ o C

Seguridad de tipos: C# garantiza que los tipos de datos variables se definan con precisión

Orientado a objetos: C# admite la creación y utilización de clases y objetos.

Orientado a componentes: C# permite a los desarrolladores reutilizar componentes existentes sin codificar desde cero

Código abierto: C# permite el desarrollo de aplicaciones en múltiples sistemas operativos, incluidos Mac y Linux

Programación estructurada: C# permite dividir los programas en funciones, lo que promueve una mejor organización.

Biblioteca enriquecida: C# ofrece una variedad de funciones integradas que aceleran el desarrollo

Velocidad: C# ofrece una rápida compilación y ejecución de código.

Facilidad de uso: pasar parámetros es sencillo en C#

**P: ¿Qué se entiende por programación orientada a objetos?**

Respuesta:

La programación orientada a objetos (POO), el enfoque utilizado por C#, consiste en diseñar software utilizando objetos que interactúan. Este tipo de programación sirve como alternativa a la programación funcional o procedimental.

**P: ¿En qué se diferencia C# de C?**

Respuesta:

Una diferencia significativa entre C# y su predecesor, C, es que C# es un lenguaje de programación orientado a objetos, mientras que C es un lenguaje de programación procedimental. Los procedimientos simplemente ejecutan una serie de pasos computacionales.

Algunas otras diferencias incluyen:

C es más adecuado para aplicaciones de hardware y programación de sistemas, mientras que C# se utiliza para aplicaciones de escritorio y móviles, así como para servicios web.

C se basa en solo 32 palabras clave diferentes, mientras que C# tiene 87

C pone mayor énfasis en las funciones, mientras que C# está más orientado al diseño.

**P: ¿Qué es una clase en C#?**

Respuesta:

Las clases son planos definidos por el usuario que se utilizan para crear objetos. Combinan de manera eficaz distintos tipos de datos en una unidad cohesiva.

Como lenguaje orientado a objetos, C# emplea clases para definir la estructura, el almacenamiento y la gestión de datos de un programa.

Dentro de una clase, hay miembros de clase conocidos como propiedades y métodos. Estos miembros pueden clasificarse como privados, restringiendo su acceso solo dentro de la clase, o públicos, permitiendo que otras partes del programa accedan a ellos y los utilicen.

**P: ¿Qué es un objeto en C#?**

Respuesta:

Un objeto hace referencia a una entidad del mundo real y es una instancia única de una clase. Por ejemplo, consideremos una clase llamada “gatos”. Dentro de esta clase, podemos tener subclases como “Tabby”, “Maine Coon” y “American Bobtail”, todas consideradas objetos.

Llevando esto más lejos, imaginemos un programa que trabaja con libros. Primero, podemos crear una clase Book que incluya propiedades como el título y el autor. Luego, al crear varias instancias de la clase Book, podemos generar varios objetos Book, cada uno con títulos y autores distintos.

Para crear un objeto, utilizamos la palabra clave “Nuevo” para construir una instancia de la clase correspondiente, lo que nos permite acceder y utilizar los métodos de clase asociados con ese objeto.

**P: ¿Qué es un método en C#?**

Respuesta:

En C#, un método es un bloque de código que contiene una serie de instrucciones que se utilizan para realizar operaciones. Los desarrolladores deben declarar métodos dentro de una clase o una estructura, y ayudan a ahorrar tiempo al reutilizar el código.

Preguntas de entrevista para desarrolladores junior de C#: más avanzadas

**P: ¿Qué es el código administrado y no administrado?**

Respuesta:

En el marco .NET, el código se divide en dos categorías: administrado y no administrado.

El código administrado se desarrolla dentro del marco .NET y se ejecuta mediante Common Language Runtime (CLR). Cualquier lenguaje escrito en .NET Framework se considera código administrado. CLR administra la memoria, la seguridad y otras tareas relacionadas con la ejecución del código administrado, y proporciona funciones de seguridad integradas.

Por otro lado, el código no administrado se desarrolla fuera del marco .NET e interactúa directamente con las funciones de bajo nivel del sistema operativo. Algunos ejemplos de código no administrado son el código heredado de VB, ASP y COM.

El código no administrado depende del sistema operativo para la administración de la memoria, la seguridad y otras tareas relacionadas con la ejecución. La responsabilidad del desarrollador es garantizar que el código no administrado sea seguro y esté protegido.

En pocas palabras, cuando el CLR ejecuta código dentro de la plataforma .NET, se considera código administrado, mientras que el código ejecutado por un programa de ejecución fuera de la plataforma .NET se clasifica como código no administrado.

**P: ¿Qué se entiende por recolección de basura en C#?**

Respuesta:

En C#, la gestión de memoria en una aplicación se realiza mediante un proceso conocido como recolección de elementos no utilizados. Este proceso automático libera de manera eficiente la memoria que ya no se utiliza, lo que permite reasignarla para nuevos propósitos.

**P: ¿Cuáles son los diferentes tipos de clases en C#?**

Respuesta:

Existen cuatro tipos de clases en C#:

Clases abstractas: proporcionan una definición común de una clase base a partir de la cual se pueden derivar otras clases. La clase consiste en una metodología o un enfoque único.

Clases estáticas: contienen elementos estáticos que solo pueden interactuar con otros elementos estáticos.

Clases parciales: permite dividir métodos, eventos y propiedades en archivos fuente .cs separados o múltiples, que luego se pueden combinar en una clase en tiempo de compilación. Se puede acceder a todos los métodos de los archivos fuente utilizando el mismo objeto.

Clases selladas: evita que los usuarios hereden de la clase y restringe el acceso a los miembros de la clase, como propiedades o métodos.

**P: ¿Cuál es la diferencia entre campos y propiedades en C#?**

Respuesta:

En el contexto de las clases y los objetos, un campo es un miembro que sirve como ubicación de almacenamiento para guardar un valor. Puede asociarse con cualquier tipo de datos.

Por otro lado, una propiedad es un miembro de la clase que facilita la lectura, escritura y cálculo del valor almacenado en un campo privado.

**P: ¿Cuáles son los diferentes tipos de declaraciones de control en C#?**

Respuesta:

Existen tres tipos principales de declaraciones de control, cada una de las cuales cumple diferentes propósitos. Entre ellos se incluyen:

Declaraciones de selección , que le permiten ramificarse a diferentes secciones de código

Declaraciones de iteración , que le permiten recorrer conexiones o realizar la misma serie de operaciones repetidamente hasta que se cumpla una condición específica.

Sentencias de salto , que permiten controlar el flujo para cambiar a otra sección del código

**P: ¿Qué son el boxing y el unboxing en C#?**

Respuesta:

En C#, los desarrolladores utilizan técnicas de conversión entre tipos de datos .NET, lo que les permite convertir tipos de referencia en tipos de valor y viceversa. Estos procesos son fundamentales para establecer una vista unificada dentro de C#.

**P: ¿Cuál es la diferencia entre String y StringBuilder en C#?**

Respuesta:

Hay dos tipos de objetos para manejar datos de texto en C#: objetos de cadena y objetos generadores de cadenas.

Los objetos de cadena son inmutables. Esto significa que no se pueden modificar una vez creados. Cualquier intento de modificar un objeto de cadena dará como resultado la creación de un nuevo objeto de cadena en la memoria.

Por otro lado, los objetos generadores de cadenas son mutables, lo que ofrece flexibilidad a los desarrolladores para modificarlos según sea necesario. Se pueden realizar operaciones como inserciones, reemplazos o añadidos en un objeto generador de cadenas. Cuando se realizan modificaciones en un objeto generador de cadenas, la cadena se actualiza en el lugar, sin necesidad de crear un nuevo espacio de memoria.