In C#,

`!= null` and `is not null` **both check for null**, but they are **not exactly the same** in terms of syntax rules, type safety, and pattern matching capabilities.

Here's the breakdown:

1. `!= null`

- **Type**: Standard inequality operator.
- **How it works**: Calls the `!=` operator for the type (which may be overloaded).
- **Null check**: Works for reference types and nullable value types.
- **Potential pitfall**: If a type overloads `!=`, the behavior might not be a pure null check.

```csharp
string s = "Hello";

if (s != null) // Uses operator !=
{
    Console.WriteLine("Not null");
}
```

2. `is not null`

- **Type**: Pattern matching syntax (introduced in C# 9).
- **How it works**: Checks if the expression **matches the pattern** `not null`.
- **Null check**: Always a pure null check — ignores any overloaded `==` or `!=` operators.
- **Extra power**: Can be combined with other patterns (e.g., type checks).

```csharp
string s = "Hello";

if (s is not null) // Pattern matching null check
{
    Console.WriteLine("Not null");
}
```

Key Differences Table

Feature	!= null	is not null
Introduced	C# 1.0	C# 9.0
Operator Overloading	Can be affected by overloaded !=	Always a pure null check
Pattern Matching	❌ No	✅ Yes
Nullable Value Types	Works	Works
Readability	Common, older syntax	More explicit in modern C#

Example Showing the Difference

```
class Weird
{
    public static bool operator !=(Weird a, Weird b) => false; //
Always false
    public static bool operator ==(Weird a, Weird b) => false; //
Always false
}


Weird w = null;


Console.WriteLine(w != null);     // false (because of overloaded
operator)
Console.WriteLine(w is not null); // true  (pure null check)
```

✅ **Recommendation**:

- Use `is not null` in **modern C#** for clarity and to avoid operator overloading pitfalls.

- Use `!= null` if you are working in older C# versions (< 9.0) or when you explicitly want to respect overloaded operators.

If you want, I can also give you a **performance comparison** between `!= null` and `is not null` in C#.
Do you want me to prepare that?