# 9. What are the default access modifiers in C#?

**Brief summary:** The default access modifier at the namespace level is internal. At the class level, it is private.

First, let's understand what a "default access modifier" means. It is an access modifier that will be applied to a type or member in case the programmer doesn't declare any access modifier explicitly.

The shortest answer to the question "What are the default access modifiers in C#?" is "**They are the most restrictive access modifiers that are valid in the given context**".

Let's start with a short reminder on access modifiers in C#:

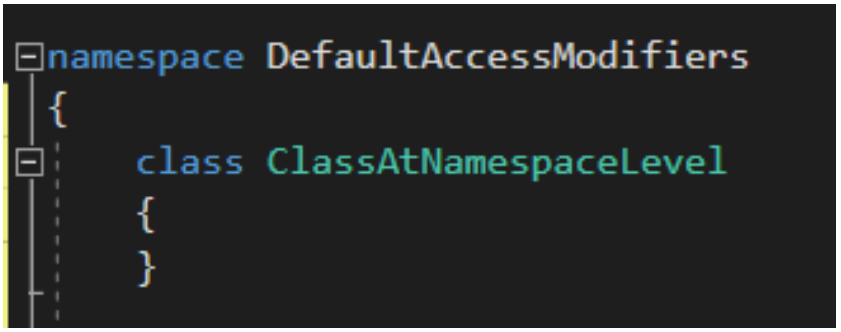
- **Public** - the type or member can be used by any other type from any assembly.
- **Internal** - the type or member can be used by any other type from the same assembly it is defined in. It can't be used outside this assembly.
- **Protected** - the type or member can be used only in the same class, or in the class that inherits from this class (no matter in what assembly)
- **Protected internal** - within the same assembly it works as internal - so it can be used by all other types. Outside this assembly it works as protected - it can be used only by types inheriting from this class.
- **Private protected** - within the same assembly it works as protected - the type or member can be used only in the declaring class or in the classes that inherit from it. Outside this assembly it can't be accessed, even by classes inheriting from this class.
- **Private** - the type or member can be used only by the code in the same class or struct.

For more information on access modifiers in C# please see the "What are the types of access modifiers in C#?" lecture.

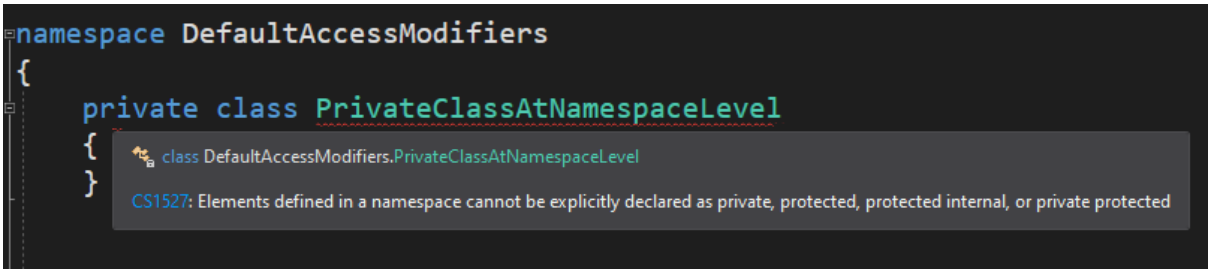
In C# there are two levels at which we can declare types or members:

- **Namespace level** - where we declare classes, structs, records, enums, delegates, and interfaces
- **Class/struct/record level** - where we declare fields, properties, methods, events, as well as inner classes, structs, records, enums, delegates, and interfaces

Let's first consider the **namespace level**. Let's have a class without any access modifier declared at namespace level:




What are the access modifiers we could use with this class? It can only be internal or public. The other access modifiers do not make any sense at the namespace level, and the compiler tells us this:




As we said before - the default access modifier is the most restrictive access modifier that is valid in the given context. In this case, internal is more restrictive than public. **That means at the namespace level the default access modifier is internal.**

Within a class it is simple - all members of a class can be declared private, and private is the most restrictive access modifier of all. **That means, the default access modifier at the class level is private**. Please note that nested classes will also be private, unlike classes declared at the namespace level.

```
//this class is internal by default
class ClassAtNamespaceLevel
{
    //this is private by default
    int number;

    //all access modifiers other than private are non-default
    public int publicNumber;


    //this class is private by default,
    //because it is declared at class level
    class InnerClass
    {

    }
}
```

**Tip: other interview questions on this topic**:

- "**What's the default access modifier for a class?**"
  *Internal, unless the class is not nested. If it is, the answer is "private".*