Introduction to Classes

With data types like `int`, `string`, and `bool` we can represent basic data and perform basic operations:

```
int count = 32;
count++;
```

What if we want to represent something more complex — something in the real world?

Say we are writing a program to manage nature reserves and we need to track forests. A forest has a certain number of trees, it has a name, it can grow, it can burn. Representing many forests with basic data types would mean tracking tons of separate variables and methods.

Instead, we can define our own custom `Forest` data type and use it like any other data type in our program. This process of bundling related data and methods into a type is called *encapsulation*, and it makes code easier to organize and reuse.

In C#, a custom data type is defined with a *class*, and each instance of this type is an *object*. This lesson will teach you how to:

- Define a class
- Add members, like properties and methods, to a class
- Customize access to those members using `public` and `private`
- Create objects from a class

In your coding adventures you may come across the term *struct*. While similar to a class it has a few differences, but we won't be covering those in this lesson.

If you do find yourself in need of a struct, you can refer to Microsoft's explanation. Structs are easy to learn after you understand classes in this lesson.

Let's get started!

☑Instructions

Here are more examples of custom classes. Testing this kind of code takes longer, so some checkpoints in this lesson will need more time to run. It's worth the wait!