Creating a strongly-typed object

In this lesson, we'll learn how TypeScript infers the types of objects and how to explicitly specify object type annotations ourselves.

Inferred object types

In the code below, what has TypeScript inferred the type of `tomScore` to be?

</> TypeScript

```
const tomScore = {
    name: "Tom",
    score: 70
}
```

Show Answer

Later in the program, if we change the `score` property value, will TypeScript be happy with this?

```
tomScore.score = 75;
```

Show Answer

What if we try to add a new property to the object later in the program?

```
tomScore.passed = true;
```

Show Answer

Using explicit object type annotations

We can explicitly specify the annotation on an object just like we would with a primitive type. Below is the `tomScore` variable with its type explicitly defined:

</> TypeScript

```
const tomScore: { name: string; score: number; } = {
    name: "Tom",
    score: 70
}
```

Wrap up

TypeScript can infer the type of an object from the assigned value. If the inferred type is not quite what we require, we can explicitly use an object type annotation.

What if we want to reuse an object type instead of redefining it each time? Is there a way to do this? Yes, there are actually several ways! We'll learn these in the following lessons.