TypeScript offers all of JavaScript’s features, and an additional layer on top of these: TypeScript’s type system. For example, JavaScript provides language primitives like string and number, but it doesn’t check that you’ve consistently assigned these. TypeScript does.

JavaScript offers a large variety of design patterns, but some make difficult for the types to be determined automatically. By using an interface, it is possible to specify the types in an object, example:

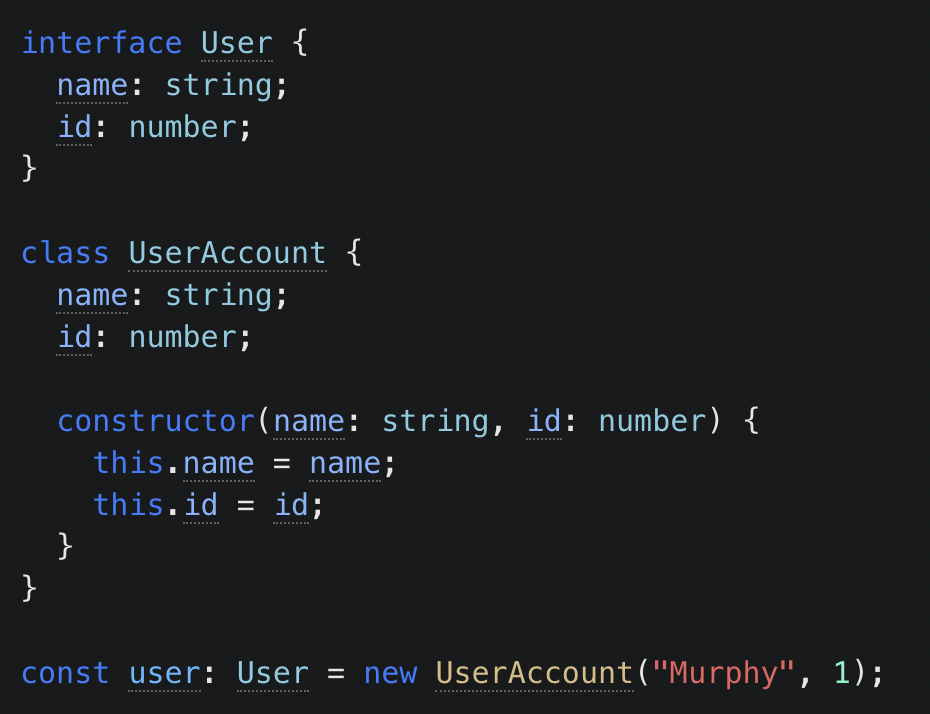
Text

Description automatically generated

Graphical user interface

Description automatically generated

Since JavaScript supports classes and object-oriented programming, so does TypeScript. You can use an interface declaration with classes:



Composing Types

With TypeScript, you can create complex types by combining simple ones. There are two popular ways to do so: with unions, and with generics.

With a union, you can declare that a type could be one of many types. For example, you can describe a Boolean type as being either true or false:

A popular use-case for union types is to describe the set of string or number literals that a value is allowed to be:

A screenshot of a computer

Description automatically generated with medium confidence