**Using Interfaces To Define Function Types**

A lesser known but nonetheless interesting feature of TypeScript interfaces is that you can also use them to define function types.

For example, you might want to define the type of a sum function that takes two numbers as input and returns their sum.

You could write this code:

type SumFn = (a: number, b: number) => number; // function type

let sum: SumFn; // making sure sum can only store values of that function type

sum = (a, b) => a + b; // assigning a value that adheres to that function type

Alternatively, you can also define the SumFn type via an interface:

interface SumFn {

(a: number, b: number): number;

}

It's up to you which alternative you prefer.

Typically, you'll encounter the first version (type SumFn) more often but it's worth knowing about the alternative, too.