How to Override Property Type In Derived Interfaces

Sometimes, an interface is created that the wrong datatype for a property.
This can be because the interface is from a third-party or because you are composing a more specific interface type and need to swap out more generic properties of the base type.

Taken from here: https://bobbyhadz.com/blog/typescript-override-interface-property

Since TypeScript is compiled to javascript, it includes a utility, called Omit, that can do the type swapping work during compilation. An Omit declaration replaces the standard extends declaration of the interface by including a list of properties to be left out.

The Omit utility type constructs a new type by removing the specified keys (as a pipe-delimited list) from the existing type.

Here is an example base interface, and a derived interface, using Omit, to change the data types of the base:

index.ts

```ts
interface Location {
  address: string;
  x: number;
  y: number;
}

interface SpecificLocation extends Omit<Location, 'address'> {
  address: {
    country: string;
    city: string;
    street: string;
  };
}

const example: SpecificLocation = {
  x: 5,
  y: 10,
  address: {
    country: 'Chile',
    city: 'Santiago',
    street: 'Example street 123',
  },
};
```

Revision #2
Created 21 February 2025 07:54:06 by glwhite
Updated 21 February 2025 08:27:28 by glwhite