TypeScript Intersection Types

Summary: in this tutorial, you will learn about the TypeScript intersection types to create a new type by combining multiple existing types.

Introduction to TypeScript Intersection types

An intersection type creates a new type by combining multiple existing types. The new type has all features of the existing types.

To combine types, you use the `&` operator as follows:

```
type typeAB = typeA & typeB;
```

The `typeAB` will have all properties from both `typeA` and `typeB`.

Note that the union type uses the `|` operator that defines a variable that can hold a value of either `typeA` or `typeB`

```
let varName = typeA | typeB; // union type
```

Suppose that you have three interfaces: `BusinessPartner`, `Identity`, and `Contact`.

```
interface BusinessPartner {
    name: string;
    credit: number;
}
```

```
interface Identity {
    id: number;
    name: string;
}

interface Contact {
    email: string;
    phone: string;
}
```

The following defines two intersection types:

```
type Employee = Identity & Contact;
type Customer = BusinessPartner & Contact;
```

The `Employee` type contains all properties of the `Identity` and `Contact` type:

```
type Employee = Identity & Contact;

let e: Employee = {
    id: 100,
    name: 'John Doe',
    email: 'john.doe@example.com',
    phone: '(408)-897-5684'
};
```

And the `Customer` type contains all properties of the `BusinessPartner` and `Contact` type:

```
type Customer = BusinessPartner & Contact;

let c: Customer = {
    name: 'ABC Inc.',
    credit: 1000000,
    email: 'sales@abcinc.com',
    phone: '(408)-897-5735'
};
```

Later, if you want to implement employee sales, you can create a new intersection type that contains all properties of `Identity`, `Contact`, and `BusinessPartner` types:

```
type Employee = Identity & BusinessPartner & Contact;

let e: Employee = {
    id: 100,
    name: 'John Doe',
    email: 'john.doe@example.com',
    phone: '(408)-897-5684',
    credit: 1000
};
```

Notice both `BusinessPartner` and `Identity` have the property `name` with the same type. If they do not, then you will have an error.

Type Order

When you intersect types, the order of the types doesn't matter. For example:

```
type typeAB = typeA & typeB;
type typeBA = typeB & typeA;
```

In this example, `typeAB` and `typeBA` have the same properties.

Summary

- An intersection type combines two or more types to create a new type that has all properties of the existing types.
- The type order is not important when you combine types.