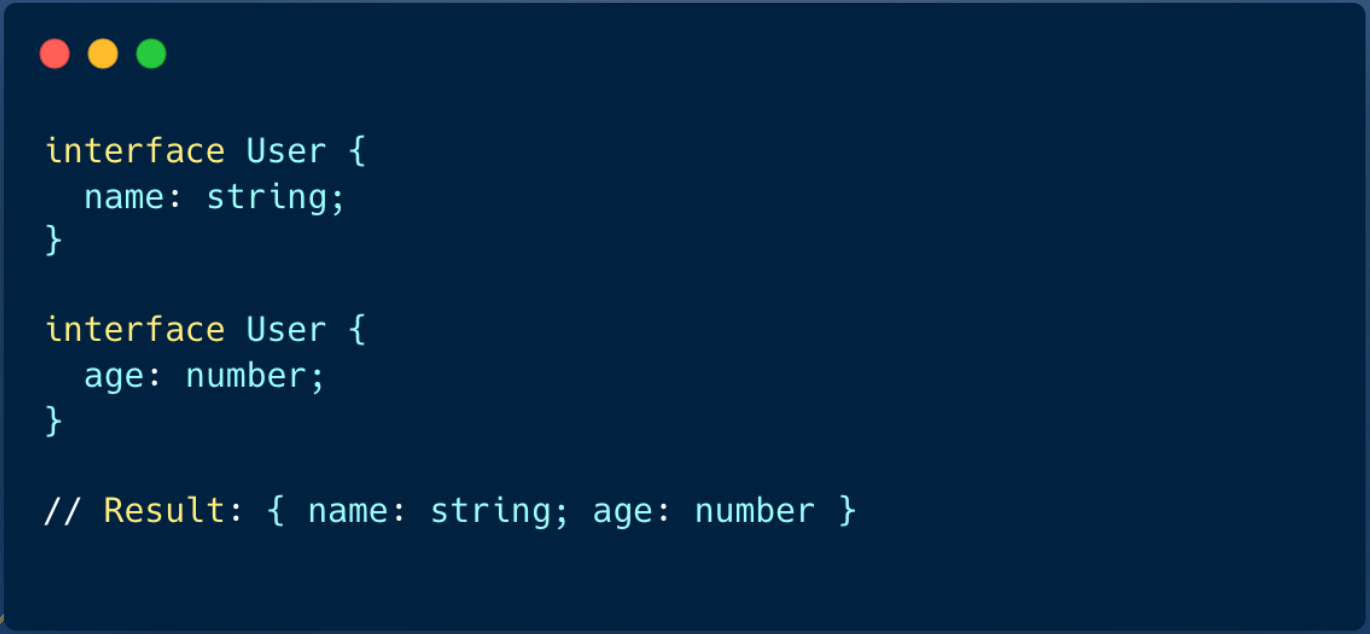
TYPE VS INTERFACE IN TYPESCRIPT

A CLEAN GUIDE TO UNDERSTAND THE KEY DIFFERENCES

DECLARATION MERGING

- Interface: Supports declaration merging
- Type: Does NOT support declaration merging

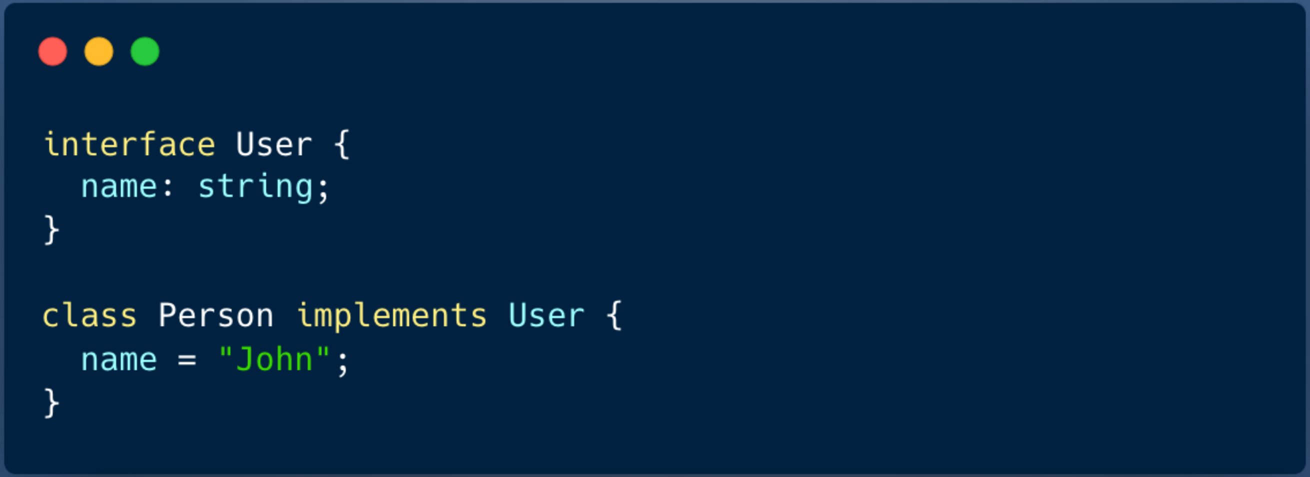


```
interface User {
  name: string;
}

interface User {
  age: number;
}

// Result: { name: string; age: number }
```

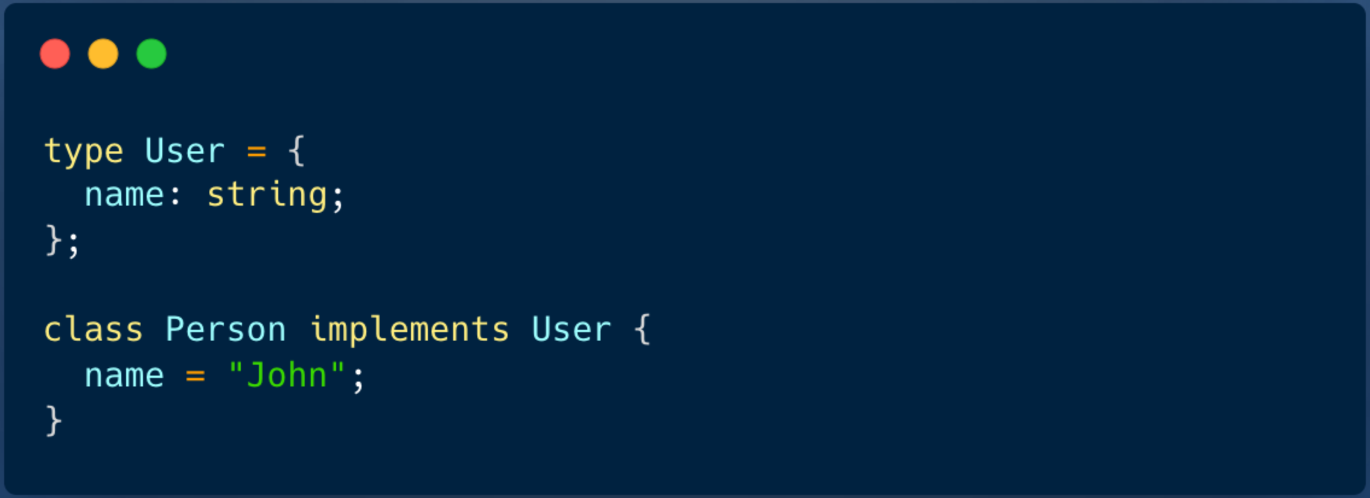
IMPLEMENTS CLAUSE

- Interfaces can be implemented by classes
- Types can also be implemented, but only object-shaped types

- Note: Unions cannot be used with implements



```typescript
interface User {
  name: string;
}

class Person implements User {
  name = "John";
}
```

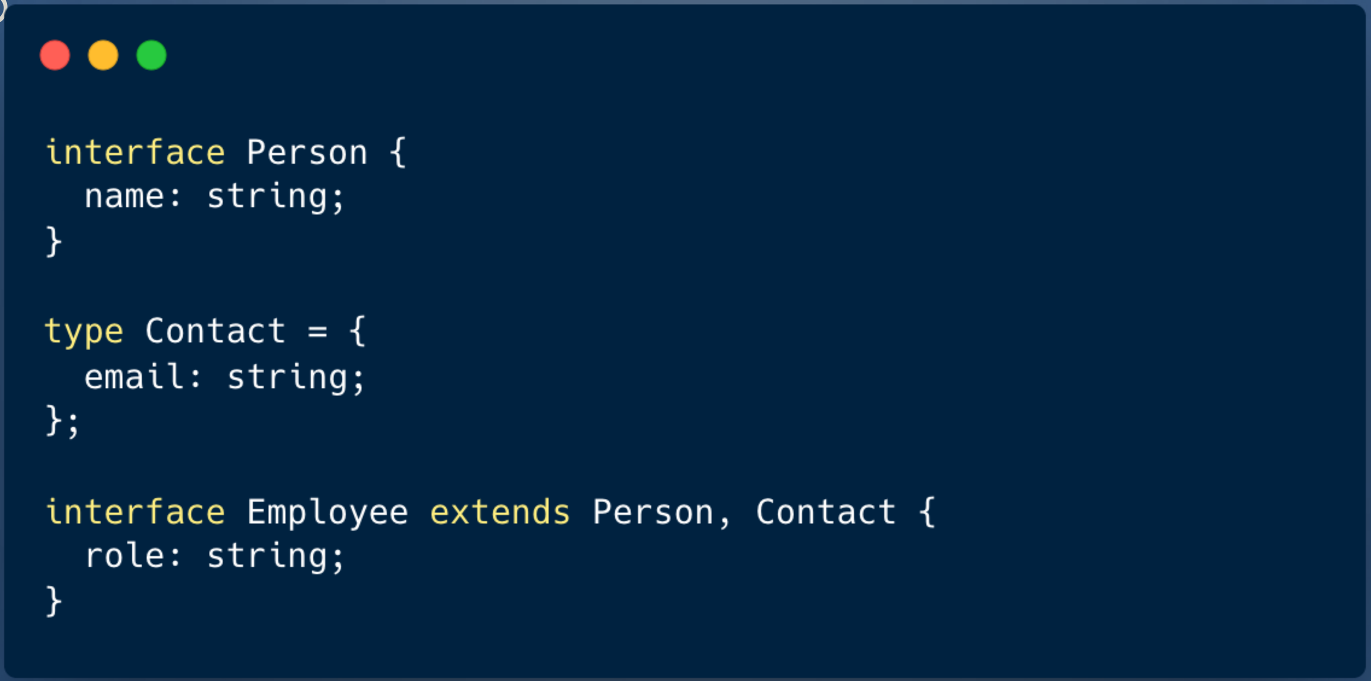


```typescript
type User = {
  name: string;
};

class Person implements User {
  name = "John";
}
```

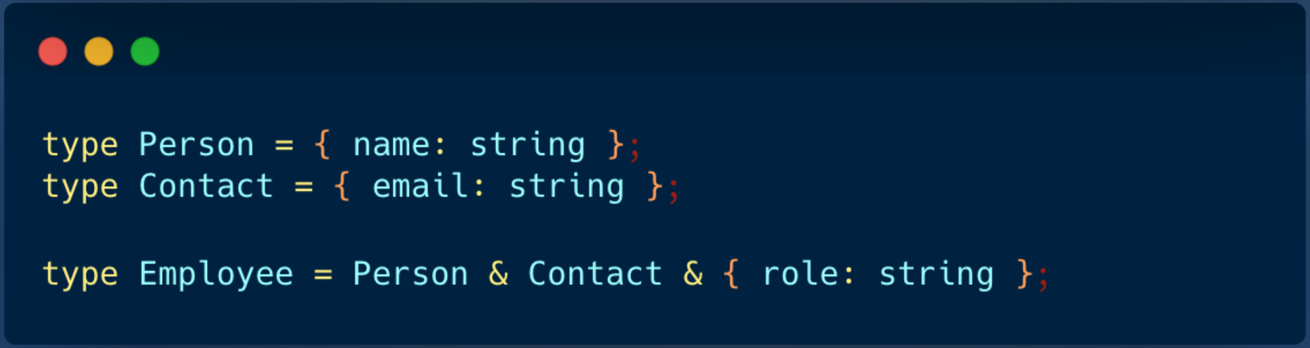
EXTENDING

- Interface: Can extend interfaces & types
- Type: Uses intersections (&) to combine

- Types offer more flexibility (e.g., extend unions)



```typescript
interface Person {
  name: string;
}

type Contact = {
  email: string;
};

interface Employee extends Person, Contact {
  role: string;
}
```



```typescript
type Person = { name: string };
type Contact = { email: string };

type Employee = Person & Contact & { role: string };
```

SUMMARY

- Declaration Merging → Interface: Yes | Type: No
- Implements → Interface: Yes | Type: Object types only
- Extending → Interface: extends | Type: intersections (&)