The Pitfall of Method Shorthand Syntax in TypeScript

Avoiding Runtime Errors with Object Property Syntax

Two Ways to Define Methods

- Method shorthand syntax
- Object property syntax
- Look similar, but behave differently
- One can lead to runtime errors!

```typescript
interface Obj {
  // Method shorthand syntax
  methodShorthand(param: string): void;

  // Object property syntax
  objectProperty: (param: string) => void;
}
```

The Problem with Method Shorthand

```ts
interface Dog {
  barkAt(dog: Dog): void;
}

interface SmallDog extends Dog {
  whimper: () => void;
}

const smallDog: SmallDog = {
  barkAt(dog: Dog) {},
  whimper() { console.log('Whimper') },
}

const brian: Dog = {
  barkAt(smallDog: SmallDog) { smallDog.whimper();  },
};

const normalDog: Dog = {
  barkAt() {},
};
brian.barkAt(smallDog);
```

```
Whimper
```

- Allows narrower parameter types
- Can lead to runtime errors
- TypeScript doesn't catch the error
- Unexpected behavior in inheritance scenarios

The Solution: Object Property Syntax

```ts
interface Dog {
  barkAt: (dog: Dog) => void;
}

interface SmallDog extends Dog {
  whimper: () => void;
}

const brian: Dog = {
  barkAt(dog: SmallDog) {},
};
```

- Stricter type checking
- Prevents narrowing of parameter types
- Catches errors at compile-time
- Safer and more predictable

Common Misconceptions

- Not related to arrow functions vs. function declarations
- Applies to both interfaces and types
- The issue is with the syntax, not the function type

```ts
interface Obj {
  methodShorthand(param: string): void;
  objectProperty: (param: string) => void;
}

function functionDeclaration(param: string) {}
const arrowFunction = (param: string) => {};

const examples: Obj[] = [
  {
    methodShorthand: arrowFunction,
    objectProperty: functionDeclaration,
  },
  {
    methodShorthand: functionDeclaration,
    objectProperty: arrowFunction,
  },
];
```

Best Practices

- Use object property syntax for method signatures
- Avoid method shorthand syntax
- Use ESLint rule: @typescript-eslint/method-signature-style

Questions?

Have you encountered sneaky runtime errors due to method syntax?

Thank You!

Remember: In TypeScript, syntax matters more than you think!