TypeScript has two narrowing-related behaviors that are both intentional. Please do not log additional bugs on this; see #9998 for more discussion.

The first is that *narrowings are not respected in callbacks*. In other words:

```
function fn(obj: { name: string | number }) {
  if (typeof obj.name === "string") {
    // Errors
    window.setTimeout(() => console.log(obj.name.toLowerCase());
  }
}
```

This is intentional since the value of `obj.name` "could" change types between when the narrowing occurred and when the callback was invoke. See also #11498

The second is that *function calls do not reset narrowings*. In other words:

```
function fn(obj: { name: string | number }) {
  if (typeof obj.name === "string") {
    console.log("Here");
    // Does not error
    console.log(obj.name.toLowerCase());
  }
}
```

This is intentional behavior, *even though `console.log` could have mutated obj*. This rule is consistently applied, even with the function is in-principle inspectable to actually have side effects

```
function fn(obj: { name: string | number }) {
  if (typeof obj.name === "string") {
    mut();
    // Does not error
    console.log(obj.name.toLowerCase());
  }

  function mut() {
    obj.name = 42;
  }
}
```