# Writeable Signals


- Creating a signal

```
readonly firstSignal = signal(42);
```

- Reading the value

```
let val = this.firstSignal();
```

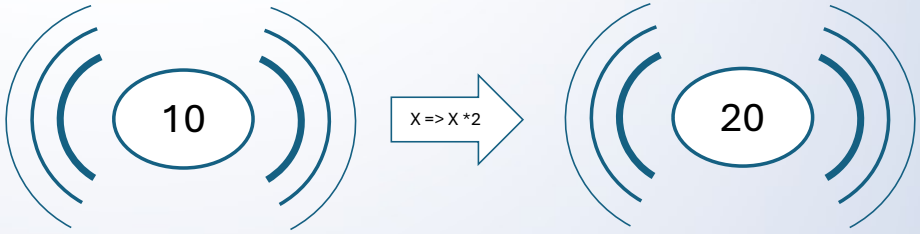
- Binding to the value

```
<h1>
    First Signal: {{firstSignal()}}
</h1>
```

- Modifying the value

```
this.firstSignal.set(10);
this.firstSignal.update(value => value + 1);
```

# Computed Signals




- Creating a computed signal

```
readonly derived = computed(() => firstSignal() * 2);
```

- Reading the value

```
let val = this.derived(); // same as writeable
```

- Binding to the value

```
<h1>
    Derived Signal: {{derived()}} // same as writeable
</h1>
```

- ~~Modifying the value~~

# Computed signals

- It's ok to
  - Use more than one signal
  - Use writeable or computed signals
  - Use constants and immutable non-signal data

```
computed(() => x() * y())
```

```
computed(() => x() * derived())
```

```
computed(() => x() * Math.PI)
```

- But don't
  - Use asynchronous code
  - Use changeable data that is not signal
  - Cause side effects
  - Modify or create other signals

```
computed(async () => x() * await calc())
```

```
computed(() => x() * Date.now())
```

```
computed(() => x() * j++)
```

```
computed(() => {
    x.update(v => v + 1);
    return x() * 10;
})
```