More on the Arrow Function Syntax

When working with **Arrow Functions**, you have a couple of *"syntax shortcuts"* available.

Most importantly, you should know about the following alternatives:

1) Omitting parameter list parentheses

If your arrow functions takes **exactly one parameter**, you may **omit** the wrapping parentheses.

Instead of

```
(userName) => { ... }
```

you could write

```
userName => { ... }
```

Please note:

- If your function takes **no parameters**, parentheses **must not be omitted** - `() => { ... }` is the **only correct form** in that case.
- If your function takes **more than one parameter**, you also **must not omit** parentheses - `userName, userAge => { ... }` would be invalid (`(userName, userAge) => { ... }` is correct)!

2) Omitting function body curly braces

If your arrow function contains **no other logic but a `return` statement**, you may **omit the curly braces** and the `return` keyword.

Instead of

```
number => {
  return number * 3;
}
```

you could write

```
number => number * 3;
```

The following code would be invalid:

```
number => return number * 3; // invalid because return keyword must also be omitted!
```
```
number => if (number === 2) { return 5 }; // invalid because if statements can't be returned
```

3) Special case: Just returning an object

If you go for the shorter alternative explained in 2) and you're trying to return a **JavaScript object**, you may end up with the following, **invalid** code:

```
number => { age: number }; // trying to return an object
```

This code would be invalid because JavaScript treats the curly braces as **function body wrappers** (not as code that creates a JS object).

To *"tell"* JavaScript that an object should be created (and returned) instead, the code would need to be adjusted like this:

```
number => ({ age: number }); // wrapping the object in extra parentheses
```

By wrapping the object and its curly braces with an **extra pair of parentheses**, JavaScript understands that the curly braces are not there to define a function body but instead to create an object. Hence that object then gets returned.