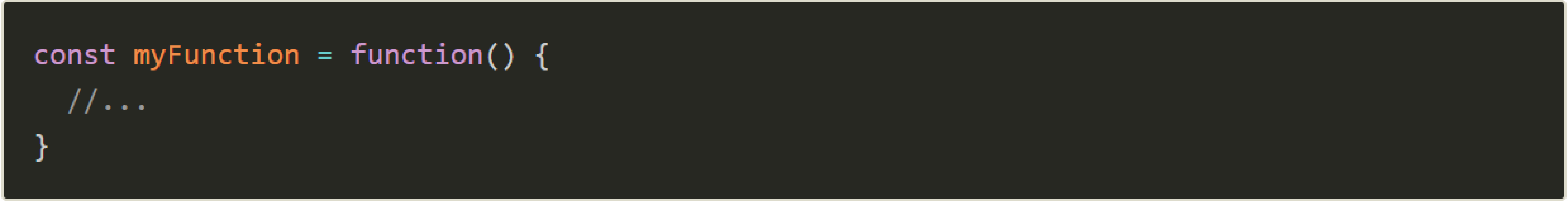
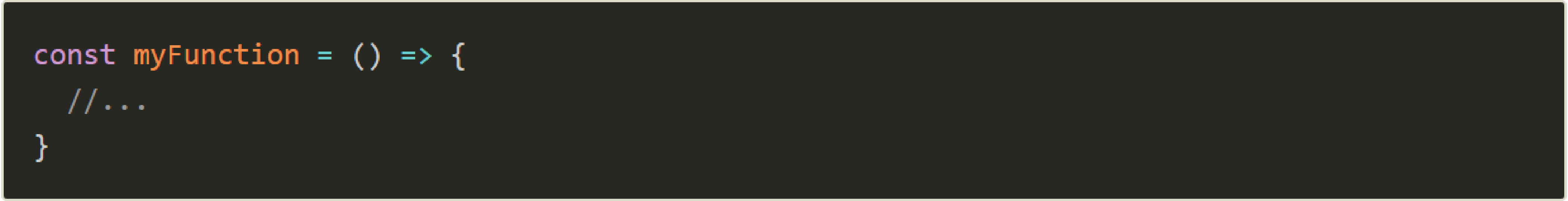
# A tutorial to JavaScript Arrow Functions

# A tutorial to JavaScript Arrow Functions

- Arrow Functions are one of the most impactful changes in ES6/ES2015, and they are widely used nowadays. They slightly differ from regular functions. Find out how
- Visually, it’s a simple and welcome change, which allows you to write functions with a shorter syntax, from:



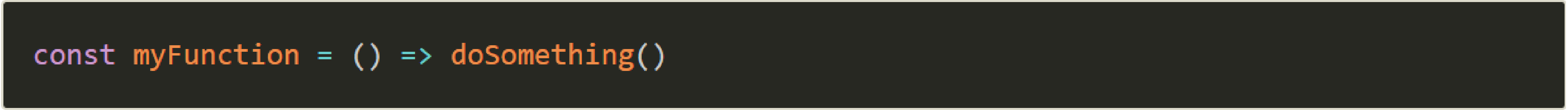
```
const myFunction = function() {
  //...
}
```



```
const myFunction = () => {
  //...
}
```

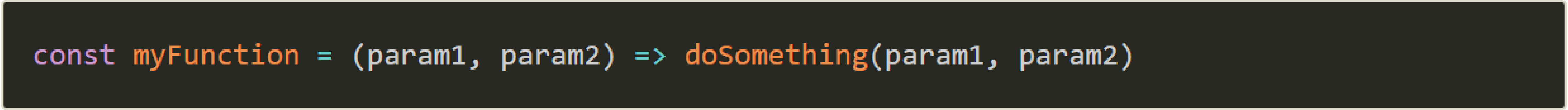
# A tutorial to JavaScript Arrow Functions

- If the function body contains just a single statement, you can omit the brackets and write all on a single line:



```
const myFunction = () => doSomething()
```

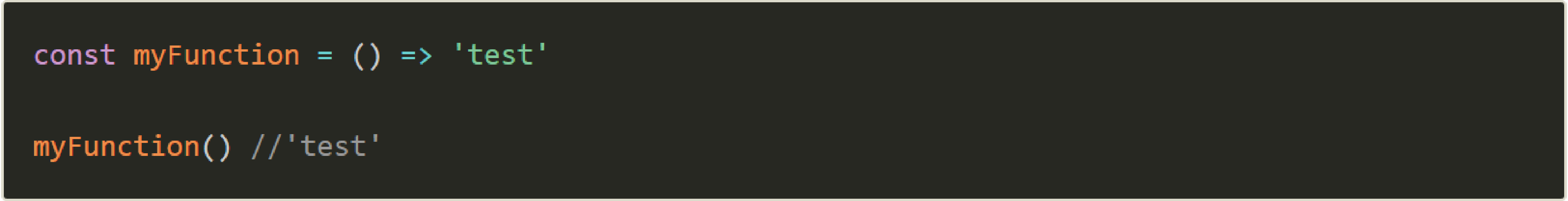
- Parameters are passed in the parentheses:



```
const myFunction = (param1, param2) => doSomething(param1, param2)
```

# Implicit return

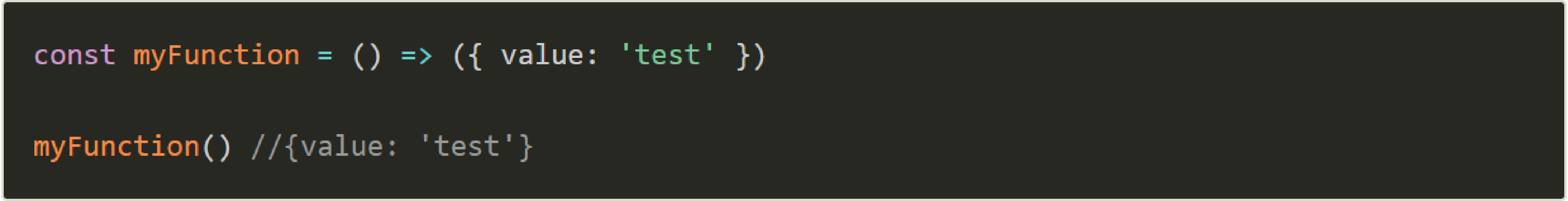
- It works when there is a one-line statement in the function body:



```
const myFunction = () => 'test'

myFunction() //'test'
```

- Another example, when returning an object, remember to wrap the curly brackets in parentheses to avoid it being considered the wrapping function body brackets:

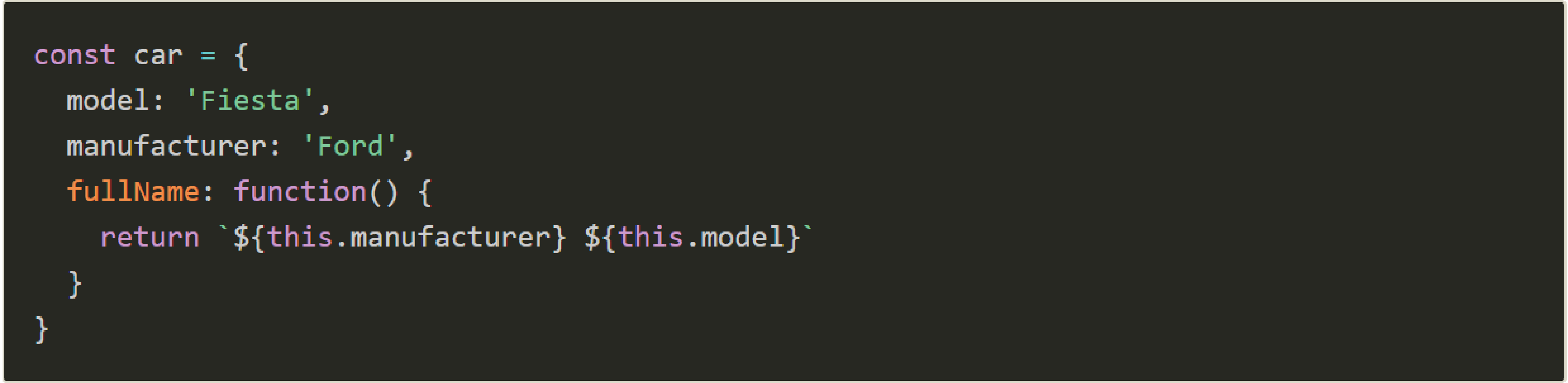


```
const myFunction = () => ({ value: 'test' })

myFunction() //{value: 'test'}
```

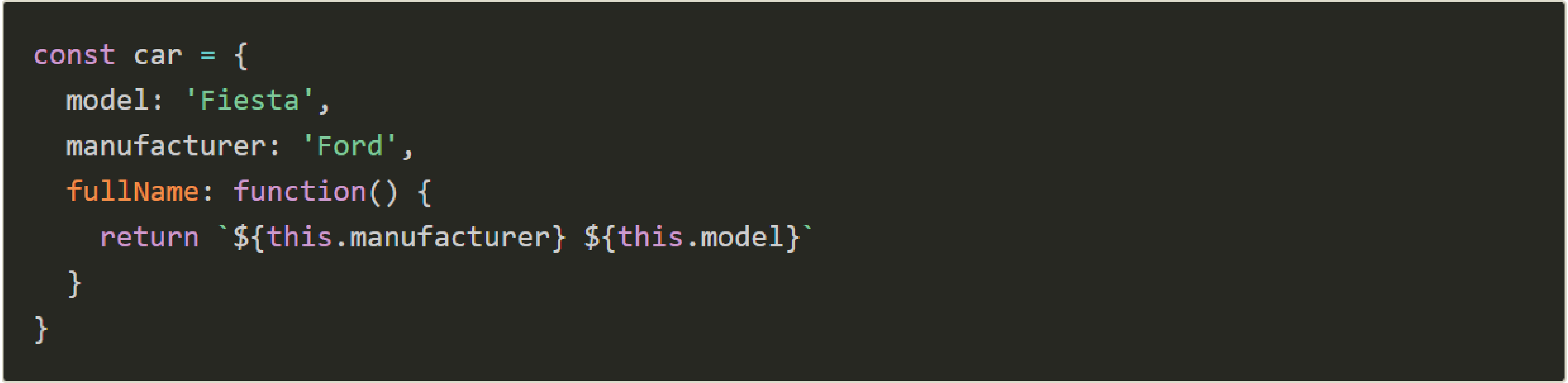
# How this works in arrow functions

- When defined as a method of an object, in a regular function this refers to the object, so you can do:



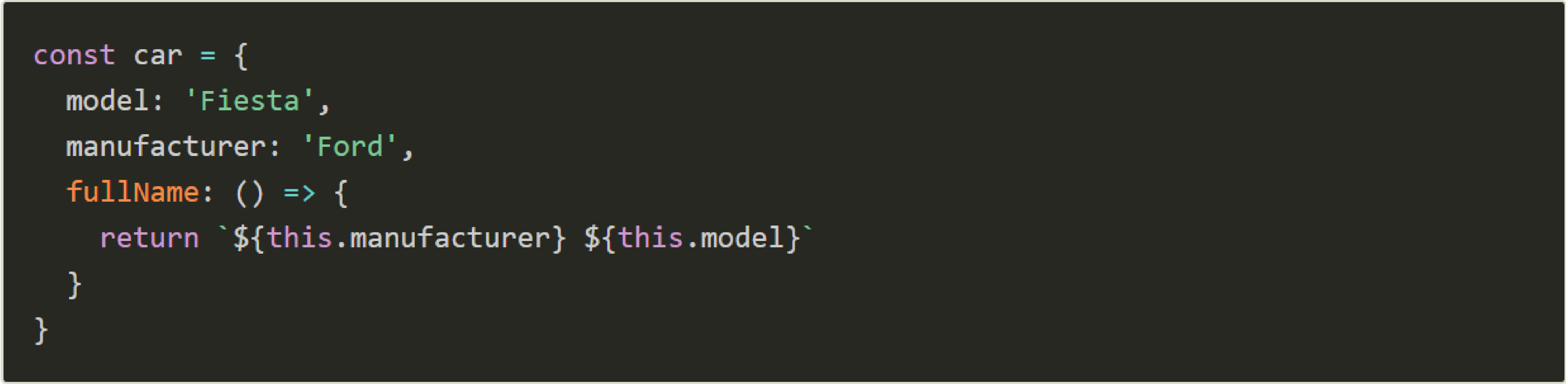
```
const car = {
  model: 'Fiesta',
  manufacturer: 'Ford',
  fullName: function() {
    return `${this.manufacturer} ${this.model}`
  }
}
```

- calling car.fullName() will return "Ford Fiesta".



```
const car = {
  model: 'Fiesta',
  manufacturer: 'Ford',
  fullName: function() {
    return `${this.manufacturer} ${this.model}`
  }
}
```

↓



```
const car = {
  model: 'Fiesta',
  manufacturer: 'Ford',
  fullName: () => {
    return `${this.manufacturer} ${this.model}`
  }
}
```