# Part 2

> @author: @Guillaume Ferron
> @date: 05.24.2019
> @version: v1.0.0

```
TLDR;

- ES 6
  - Arrow function
  - Strings
```

---

## ES 6

### Arrow Function

**A big issue that comes up with Javascript is the context i.e, the clustering of scopes. If you have a variable in a function, it will be only accessible in that function. On the other hand, if you have a variable declared globally, it will be accessible in that same function but in all other functions as well.**

One special case that we end up encountering very often is the `this` context sharing. For the same reason as mentioned above, `this` represents the context of the instance you're using it in. But what if you want to use that same context in a callback function ?

That is what you usually end up doing :

```
const _this = this;
console.log('1. - ' + this);

doSomethingAsynchronous(function(param) {
  console.log(_this); // will log the same as 1.
  console.log(this); // will log something different than 1.
})
```

In the case above, `_this` is a variable we put the current `this` context so that it is fixed. It is for the reason that calling `this` inside the callback function will be different since the context is different.

Though, it ends up being tedious and can get a bit confusing if you chain callbacks.

What ES6 brings to the table is the arrow functions, where it passes the context directly in the function. The exact equivalent of the example above will then be:

```
console.log('1. - ' + this)

doSomethingAsynchronous((param) => {
  console.log(this); // will log the same as 1.
})
```

`() => {}` is a shorthand syntax that ends up being very useful. It also allows to shorten any type of function declaration by striping out the `return` syntax, see below.

```
// ES5
let addOne = function (param) {
  return param + 1;
}
```

```
// ES6
let addTwo = param => param + 1;
```

---

## Strings

**This one is a very small change but is, to my opinion, clearer and cleaner.**

When you want to include variables in a string, you usually do that:

```
const str = 'My variable 1 is ' + variable1 + ' and my variable 2 is ' + variable2;
```

You can shorten it a little bit by using the ES6 syntax that uses `${}` and backquotes `` ` ``

```
const str = `My variable 1 is ${variable1} and my variable 2 is ${variable2}`
```

---