# HOW THE THIS KEYWORD WORKS

👉 **this keyword/variable:** Special variable that is created for every execution context (every function). Takes the value of (points to) the "owner" of the function in which the `this` keyword is used.

☝ `this` is **NOT** static. It depends on **how** the function is called, and its value is only assigned when the function **is actually called**.

**EXECUTION CONTEXT**

- ✅ Variable environment
- ✅ Scope chain
- 👉 this keyword

it is on 3rd of execution c.
i.e executed after above 2

ways to call function →

- **Method** 👉 `this` = <Object that is calling the method>
- **Simple function call** 👉 `this` = `undefined`
  not attached to any object
  In strict mode! Otherwise: `window` (in the browser)
- **Arrow functions** 👉 `this` = <`this` of surrounding function (lexical `this`)>
  not allowed to have own this keyword
  Don't get own `this`
- **Event listener** 👉 `this` = <DOM element that the handler is attached to>
- other ways to call function
  **`new`, `call`, `apply`, `bind`** 👉 *<Later in the course... ⌛>*

☝ `this` does **NOT** point to the function itself, and also **NOT** the its variable environment!

👉 Method example:

```javascript
const jonas = {
  name: 'Jonas',
  year: 1989,
  calcAge: function() {
    return 2037 - this.year
  }
};
jonas.calcAge(); // 48
```

`calcAge` is method

`jonas`
" value of this here

`1989`

Way better than using `jonas.year`!