How does the 'this' keyword work?

Built in keyword in object oriented languages

Will it work the same in Javascript?

No, it's quirky

In theory it's a keyword that's automatically 'bound' (points to) the relevant object when a method is called

'this' inside an object's function (method)

```
var rabbit = {
  heightAboveGround: 0,
  jump: function(howHigh){
    this.heightAboveGround += howHigh;
  }
}


rabbit.jump();

rabbit.heightAboveGround
```

But `this` is always available

I can console log `this` in the first line of my code

```
console.log("hello world");
console.log(this);
```

The global object

- Wherever JS runs (in the browser, or on the server with Node.js) there's a special object called the global object
  - its properties include built in objects like Math and String
  - in the browser it's called the Window object

'this' inside a regular function

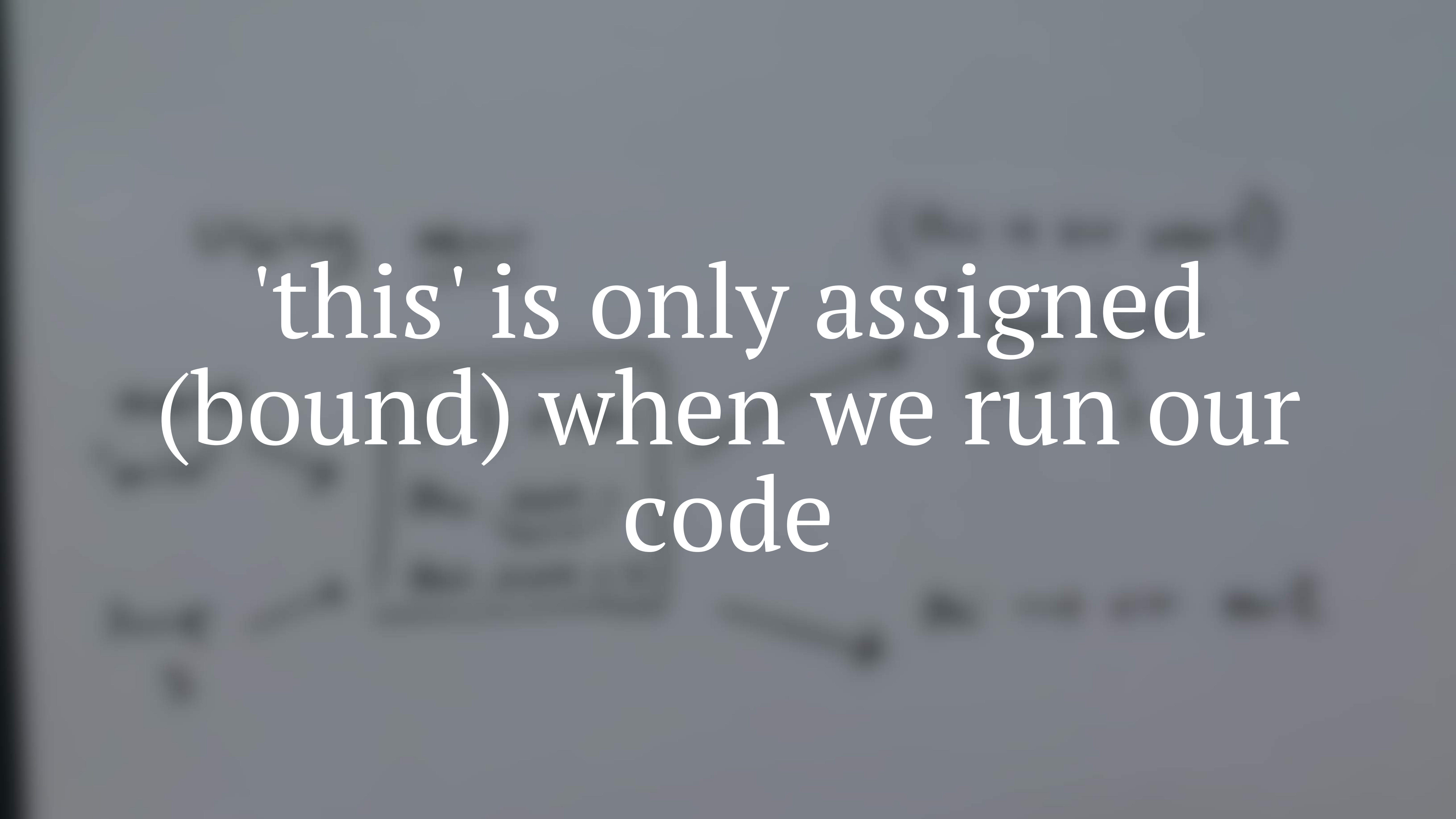
```
function multiplyBy2 (num) {
    console.log(this);
}
multiplyBy2(3);
```

What is logged?

'this' in a Constructor function

```
function User (name, score){
  this.name = name;
  this.score = score;
}

var user1 = new User("Will", 3);
var user2 = new User("Max", 5);
```



'this' is only assigned (bound) when we run our code

`this` always points to what Javascript thinks is the relevant object

Useless

```
console.log(this) <- Global object (Window)

function multiplyBy2 (num) {
    console.log(this);
}
multiplyBy2(3); <- logs Global object (Window)
```

Completely Vital!

```
rabbit.jump()` <- 'this' is rabbit

var user1 = new User("Will", 3);` <- 'this' is the new user we created
```