this

Class	
Created	@Jan 16, 2021 4:42 PM
Materials	
Reviewed	☐
Type	

1-When a function that is a property of an object is called by referencing the function as a property of that object, `this` in the function body refers to the object to which the function belongs.

2-This is the case even if the function is a property of a prototype that the object is inheriting properties from.

```
function City(nickname) {
    this.nickname = nickname;
}

City.prototype.welcomeTo = function() {
    console.log('Welcome to ' + this.nickname);
};

new City('the Big Apple').welcomeTo();
```

GLOBAL OBJECT PROB.

The value that this refers to within a function is determined when the function is called (in this way, this is like a parameter).

In such cases the value that this refers to is the global object.

If you are using strict mode, this will not refer to the global object but will instead be undefined.

THIS USE

*the value of this in a variable that will be accessible to the nested function.

```
function City(nickname) {
    this.nickname = nickname;
}

City.prototype.welcomeTo = function() {
    console.log('Welcome to ' + this.nickname);
};

City.prototype.waitThenWelcomeTo = function() {
    var city = this;
    setTimeout(function() {
        city.welcomeTo();
    }, 1000);
};
```

*bind method

```
City.prototype.waitThenWelcomeTo = function() {
    setTimeout(function() {
        this.welcomeTo();
    }.bind(this), 1000);
};
```

*call and apply

call takes any number of other arguments which will all be passed to the function.

apply allows you to use an array containing all the arguments to pass.