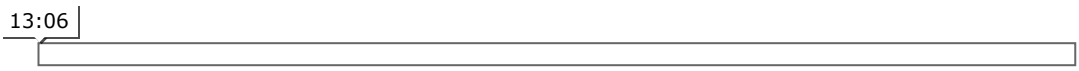
Watch: ⚡ **Aplicando novedades de la API de Array: desde ES2015 hasta ES2020**



Tienes el código de este vídeo en GitHub.

En ES5, ya teníamos dos formas de iterar un array: el loop `for` y `forEach`:

```
for (let i = 0; i < checkboxes.length; i++) {
  selectedValues.push(checkboxes[i].value);
}
```

```
checkboxes.forEach(function (checkbox) {
  selectedValues.push(checkbox.value);
});
```

En ECMAScript 2015 nos llegó `for ... of`, una manera de escribir loops más fácilmente, sin tener que iterar por el índice:

```
for (const checkbox of checkboxes) {
  selectedValues.push(checkbox.value);
}
```

Escogeremos según `for...of` y `forEach` según si necesitamos hacer un early return. Es decir, el return de este código sólo parará la ejecución del caso en que se cumpla el if, pero seguirá iterando por el resto del array:

```
list.forEach(function (item) {
  if (checkCondition(item)) {
    return;
  }

  show(item);
});
```

En cambio, si usamos `for...of`, un return parará la ejecución del loop y no seguirá iterando aunque queden elementos en el array:

```
for (const item of list) {
  if (checkCondition(item)) {
    return;
  }

  show(item);
};
```

En ES2016 llegó el método `includes`, que nos permite validar si un elemento existe en un array, lo que antes solíamos hacer con `indexOf`:

```
function isInList(item, list) {
  return list.indexOf(item) !== -1;
```

```
}

// ES2016
function isInList(item, list) {
  return list.includes(item);
}
```

Finalmente, en ES2019 llegó `flat`, que nos permite desempaquetar arrays anidados en un nuevo array:

```
const list = [0, 1, 2, [3, 4]];

list.flat() //  [0, 1, 2, 3, 4]
```