

Day 20, Today's Topic

# Destructuring Part 7

DAILY  
JS

```
/* ===== */
/* ===== Daily JS - Day 20 ===== */
/* ===== */
```

## # Day 20: Destructuring Part 7

Alright, you must be thinking

> Hey Madhav, enough with this destructuring stuff, please move on with it, tell me the next topic.

Alrightttt, I swear it's the last, and from tomorrow we will see some other topics..

Let's see what all did we do in destructuring till now...

1. Pulling properties out of objects
2. Pulling properties out of objects when they are passed as function arguments
3. Pulling values out of arrays
4. To pull out a property inside an object which is inside an array
5. To pull out an array element which is inside a property of an object
6. Working with complex objects.

Today's topic is going to be the advantage of destructuring properties of objects as functional arguments.

```
/* ===== madhavbahl.tech/dailyjs/day20 ===== */
/* ===== Join Discord: madhavbahl.tech/discord-c2e ===== */
```



```
/* ===== */
/* ===== Daily JS - Day 20 ===== */
/* ===== */
```

## ## 7. Advantage of destructuring properties of objects ## as function arguments

Yes, I won't bore you today.  
Our topic today will be very short and concise.

Apart from what I am going to discuss, there are some other benefits of destructuring that you can easily find on the internet, like improving the readability of the code, making things a little bit easier for the programmer by destructuring the imports at the top of the script (which, of course, removes some amount of confusion which may appear at later stages)

Here's one thing that I totally love about destructuring, it can help you deal with the messy function arguments.

To understand this, let's see an example

```
/* ===== madhavbahl.tech/dailyjs/day20 ===== */
/* ===== Join Discord: madhavbahl.tech/discord-c2e ===== */
```

```
/* ===== */
/* ===== Daily JS - Day 20 ===== */
/* ===== */
```

### ## Example

**\*\*Let's say we have a huge code base, and somewhere in the middle we have a function which registers the events\*\***

```
```js
// In a huge code base, somewhere there exists a function called registerEvent
function registerEvent (eventName, eventDate, venue, timings,
                        organizerName, organizerEmail) {
    // Create a new event
}
```
```


Somewhere in the middle we need to call it.  
BUT, to call it we must enter the order of arguments corerctly.

With such a huge code base, it is unlikely that we would remember, so go back to the function definition and see the order of arguments.

```
```js
// Other Code
// Other Code
// Other Code
// Other Code

registerEvent ('DanceFest', '10 oct', 'Delhi', '9 to 12', 'Ram', 'ram@xyz.com');
```
```

```
/* ===== madhavbahl.tech/dailyjs/day20 ===== */
/* ===== Join Discord: madhavbahl.tech/discord-c2e ===== */
```



```
/* ===== */
/* ===== Daily JS - Day 20 ===== */
/* ===== */
```

### ## What if it is needed again?

Somewhere in the middle we need to call it, but then we need to remember the order of args, again we would need to go back and check.

```
```js
  // Other Code
  // Other Code
  // Other Code
  // Other Code
  // Other Code

  // Now we need to call it again
  registerEvent ('... ', '... ', '... ', '... ', '.... ', '... ');
```
```

It's becoming challenging to remember the order of arguments.

**\*\*Solution? Pass an object which contains all those properties.\*\***

```
/* ===== madhavbahl.tech/dailyjs/day20 ===== */
/* ===== Join Discord: madhavbahl.tech/discord-c2e ===== */
```



```
/* ===== */
/* ===== Daily JS - Day 20 ===== */
/* ===== */
```

### ## Solution to this problem - Destructuring

We can simply remove this confusion by wrapping all the details in an object and passing that object in the argument (destructuring the same in function arguments)

```
```js
const eventDetails = {
  eventName: 'DanceFest',
  eventDate: '10 oct',
  venue: 'Delhi',
  timings: '9 to 10',
  organizerName: 'Ram',
  organizerEmail: 'ram@xyz.com'
};

function registerRefactored ({eventName, eventDate, venue,
                             timings, organizerName, organizerEmail}) {
  // Create a new event

  // Here, the order of arguments doesn't matter,
  // the properties gets destructured from the object
  // You just have to pass the object while calling the function
  signupRefactored (eventDetails);
}
```
```

That's it for today, I'll be back tomorrow with another interesting JS concept.

```
/* ===== madhavbahl.tech/dailyjs/day20 ===== */
/* ===== Join Discord: madhavbahl.tech/discord-c2e ===== */
```

```

/* ===== */
/* ===== Daily JS - Day 20 ===== */
/* ===== */

// In a huge code base, somewhere there exists a function called registerEvent
function registerEvent (eventName, eventDate, venue, timings, organizerName, organizerEmail) {
  // Create a new event
}

// Other Code
// Other Code
// Other Code
// Other Code
// Other Code

// Somewhere in the middle we need to call it
// BUT, to call it we must enter the order of arguments corerctly
// With such a huge code base, it is unlikely that we would remember,
// So go back to the function definition and see the order of arguments.
registerEvent ('DanceFest', '10 oct', 'Delhi', '9 to 12', 'Ram', 'ram@xyz.com');

// Other Code
// Other Code
// Other Code
// Other Code
// Other Code
// Other Code

// Now we need to call it again
registerEvent ('... ', '... ', '... ', '... ', '....', '... ');

// It's becoming challenging to remember the order of arguments
// Solution? Pass an object which contains all those properties
const eventDetails = {
  eventName: 'DanceFest',
  eventDate: '10 oct',
  venue: 'Delhi',
  timings: '9 to 10',
  organizerName: 'Ram',
  organizerEmail: 'ram@xyz.com'
};

function registerRefactored ({eventName, eventDate, venue, timings, organizerName, organizerEmail}) {
  // Create a new event
}

// Here, the order of arguments doesn't matter, the properties gets destructured from the object
// You just have to pass the object while calling the function
signupRefactored (eventDetails);

/* ===== madhavbahl.tech/dailyjs/day20 =====
## ===== Join Discord: madhavbahl.tech/discord-c2e =====
*/

```

# Thank you!

**Feel free to reach out...**

Email: [theleanprogrammer@gmail.com](mailto:theleanprogrammer@gmail.com)

Web: [madhavbahl.tech/](http://madhavbahl.tech/)

Github: [github.com/MadhavBahlMD](https://github.com/MadhavBahlMD)

LinkedIn: [linkedin.com/in/madhavbahl/](https://linkedin.com/in/madhavbahl/)

Insta: [instagram.com/theleanprogrammer/](https://instagram.com/theleanprogrammer/)

**DAILY**  
**JS**