Built-in JavaScript Functions

Adhithi Ravichandran

Software Consultant | Author | Speaker

@AdhithiRavi | www.adhithiravichandran.com

Functional Way in JavaScript Functions

Map

Filter

Flat

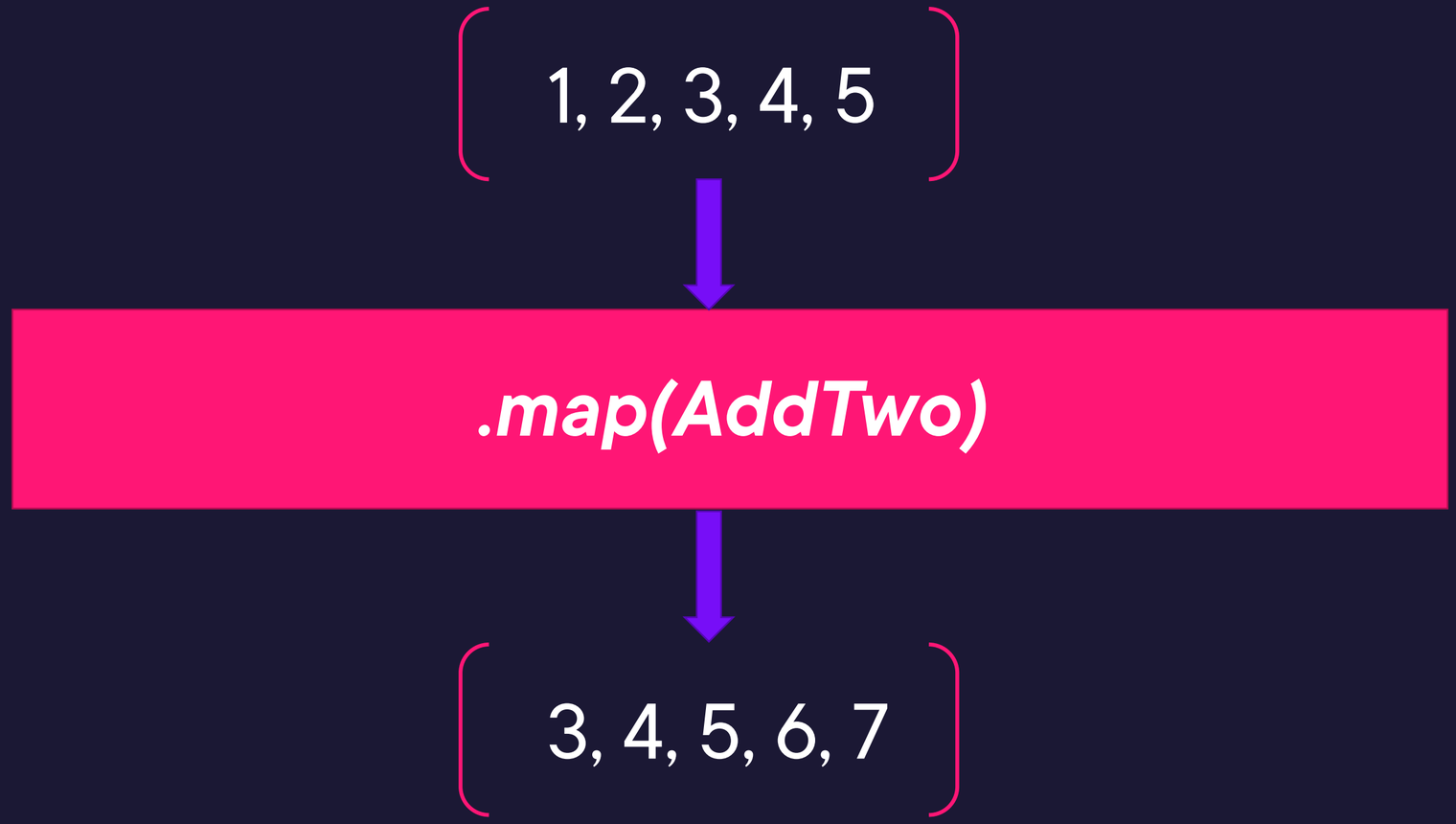
Reduce

Functional Programming Mindset – Don’t worry about **how** the problem is solved, instead focus on **what** the problem is.

Map

The *map()* function creates a new array populated with the results of calling a provided function on every element in the calling array.

Map



Filter

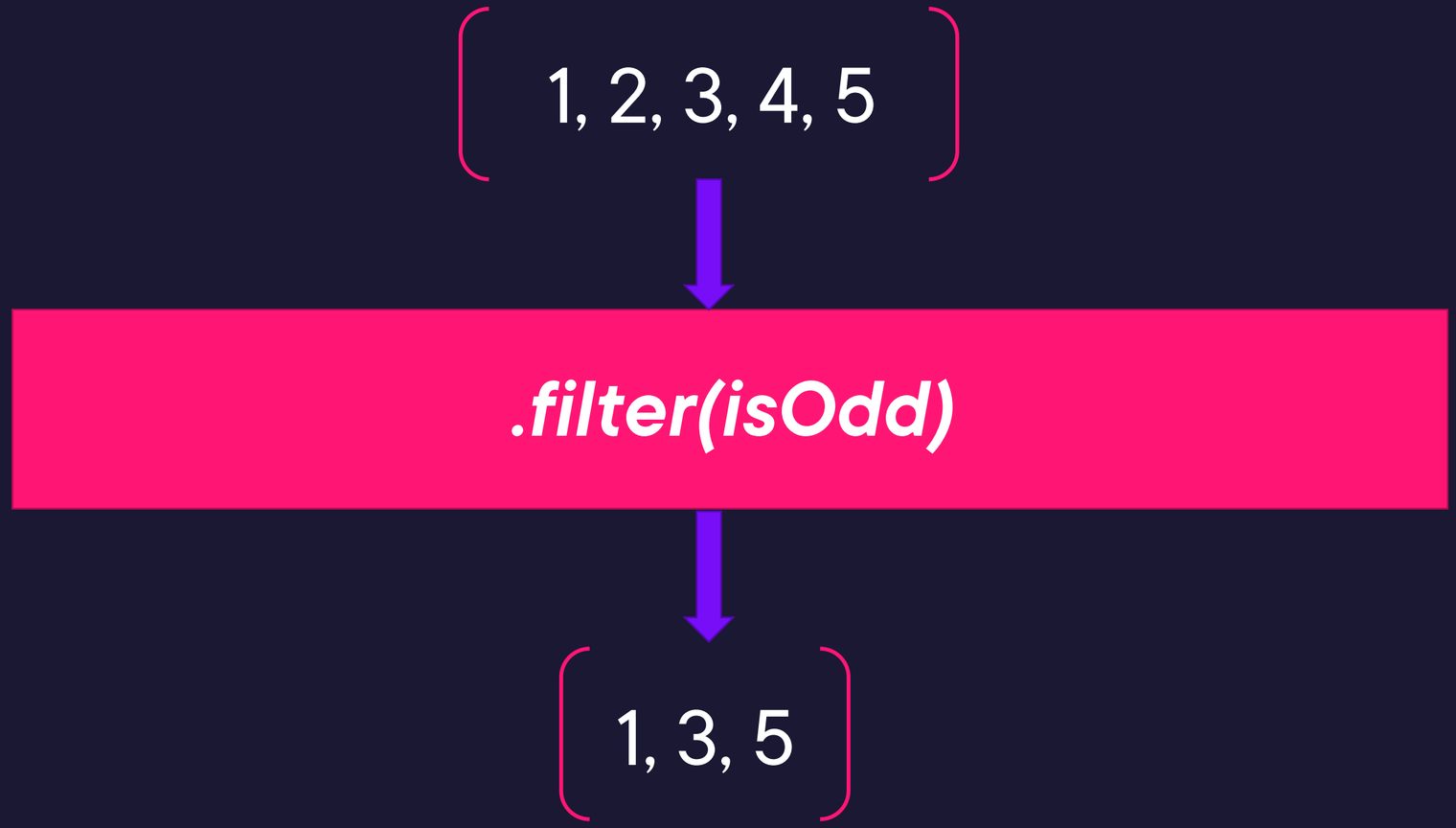
filter() Function

Creates a shallow copy of a portion of a given array.

Filters down to just the elements that pass the callback func test.

Array elements which do not pass the callback func test are not included in new array!

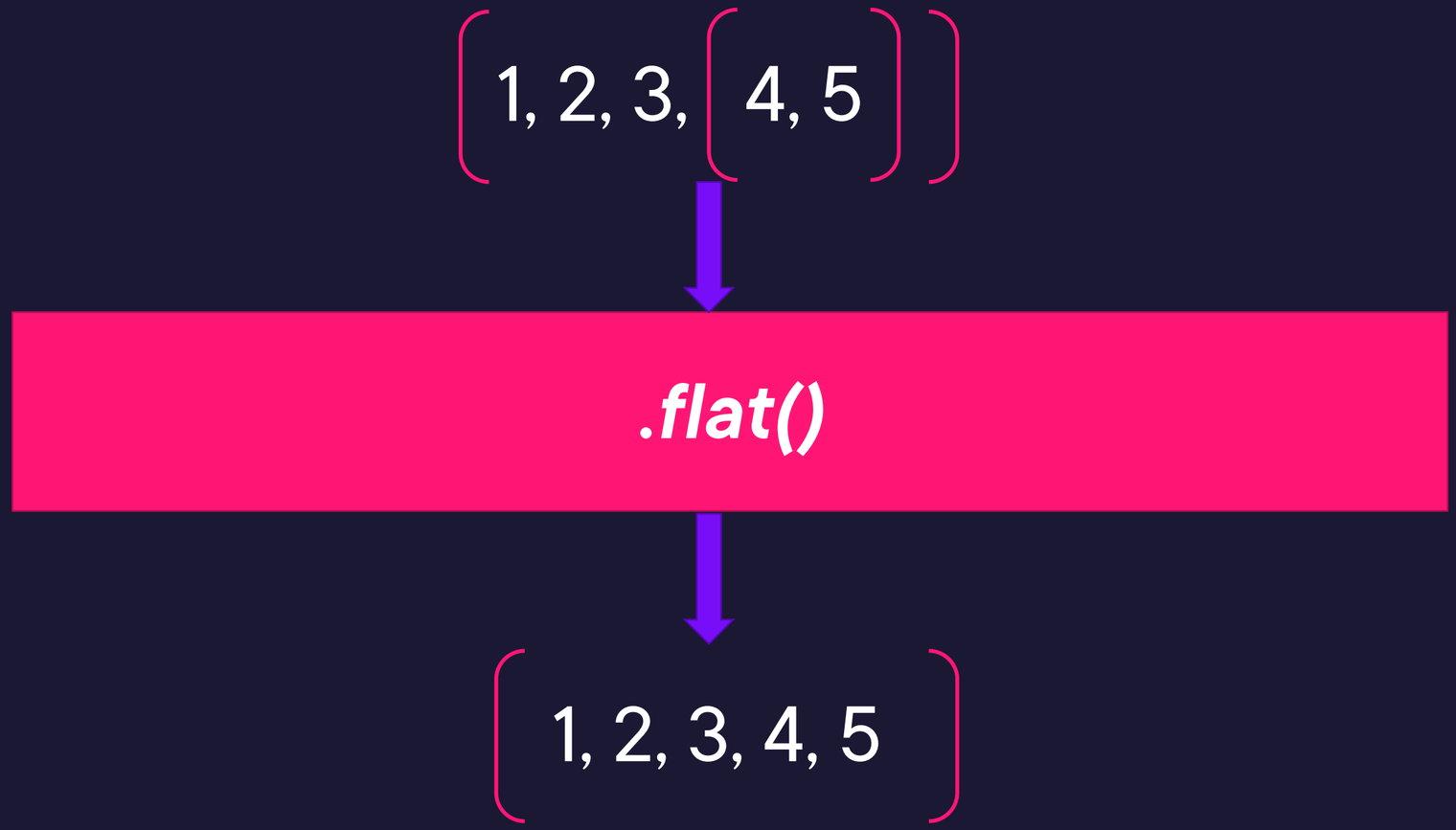
Filter



Flat

The *flat()* function creates a new array with all sub-array elements concatenated into it, recursively up to the specified depth.

Flat

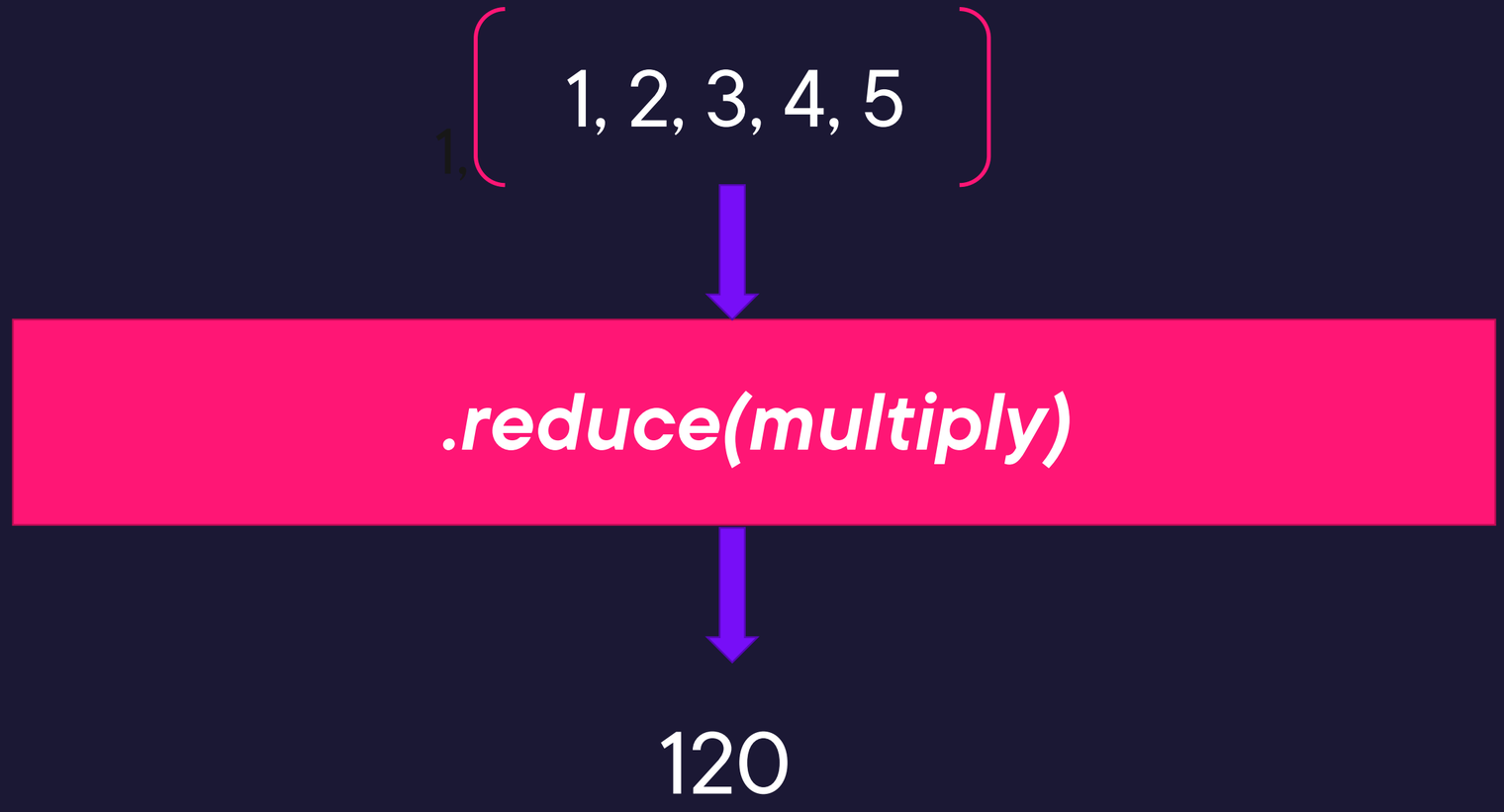


Reduce

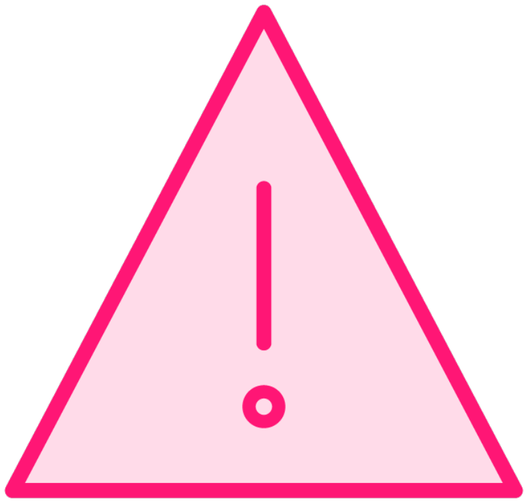
Reduce

1. Runs a reducer callback function over all elements in array
2. Runs in ascending-index order, and accumulates them into a single value
3. Reduces an array to a single value

Reducers



reduce() is a central concept in FP. It does not mutate the array on which it is called. But function provided as callbackFn can.



Reduce can be difficult to understand

Developers must weigh in readability tradeoffs vs benefits of reduce

Clear documentation to help with readability

Summary

References and More Information

- https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Global_Objects/Array

- https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Global_Objects/Function

Recap

Built-in Functional JS:

- Map
- Filter
- Flat
- Reduce

Up Next:

Advanced Concepts of Functional Programming