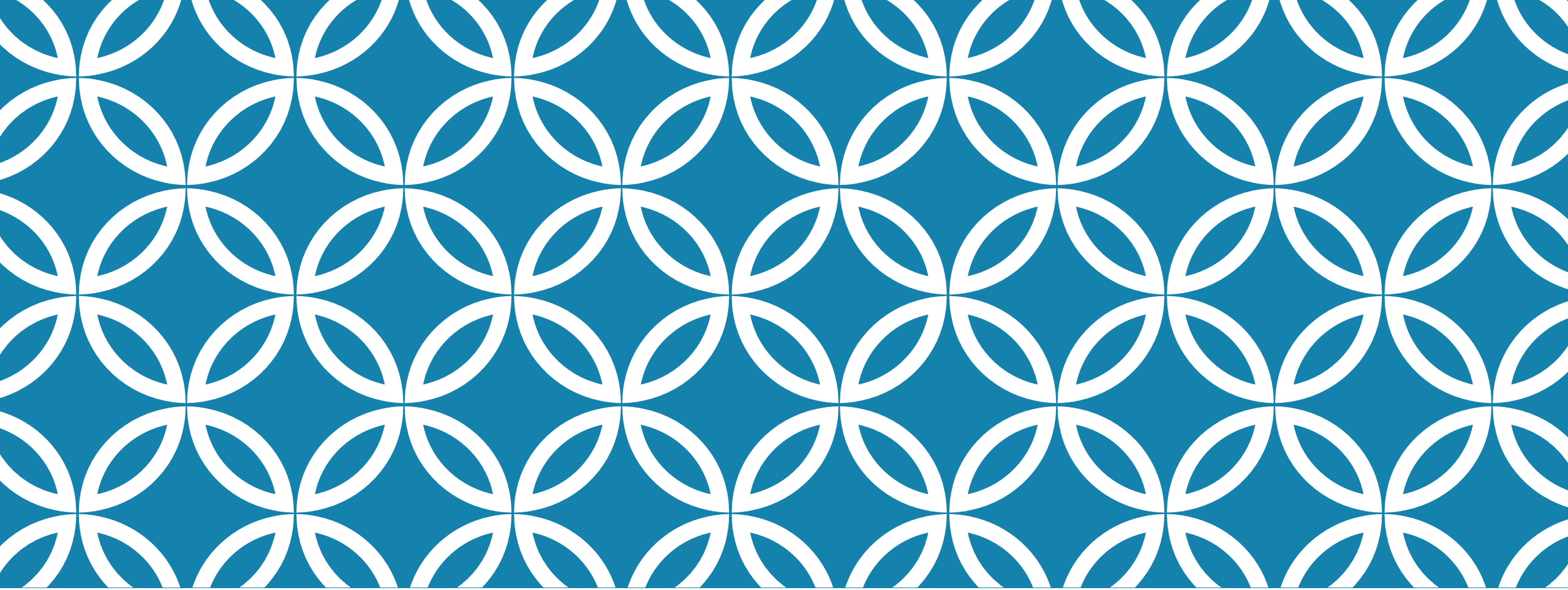

FUNCTIONAL PROGRAMMING IN JAVASCRIPT

By Prashanth Puranik
Web Developer and Trainer

Brought to you by
SkewCode
www.skewcode.com

WHAT IS FUNCTIONAL PROGRAMMING (FP)?

A Programming Paradigm - just like Procedural Programming, OOP etc.

FP is a type of declarative paradigm for programming

Existed since 1950s. Has its base in **Lamda Calculus** – a model of computation (like Turing machine)

Gaining popularity of late – especially with the advent of big data etc.

EXAMPLES OF FP LANGUAGES

Functional Programming (FP) Languages

- Lisp – amongst earliest FP languages (1950s)
- Scheme – heavily influenced the design of JavaScript
- Haskell – used in academia
- F# - by Microsoft

Programming Languages that support FP style programming (for the most part)

- Python
- R
- JavaScript
- Scala

ADVANTAGES

- **Easier to test**
  - Pure functions make inputs for a function explicit and leave state and inputs unchanged
- **Easier to reason about and understand**
  - “Lambdas” in functional programming are pure functions
  - In FP-style, we specify what is to be done, not how (declarative programming)
  - They do one specific thing and do it well
- **Easier to compose larger functionality from smaller building blocks**
  - Functions being first-class citizens in FP languages, can be passed to other functions and returned from them
  - This allows composing functions to get desired functionality
  - Also functions in FP languages support chaining of function calls

LET'S GET SOME THINGS CLEAR

It is NOT a replacement for other paradigms.

- It isn't a new paradigm
- In languages that support FP constructs, FP-style code can co-exist with code involving classes and objects (OOP)

It does not simply mean programming using functions – that is imperative programming

CONCEPTS

- **Pure Functions**
  - Do not mutate state – no change in arguments or global state
  - They accept all inputs explicity as arguments
  - Give same output for same inputs – do not use global state within
  - Are functions in the mathematical sense of the word (rather than in a programming sense)

- **Higher-order functions**
  - Functions that accept other functions as input (arguments), or,
  - Functions that return other functions
  - Note: In JavaScript, functions are objects – hence first-class citizens. Functions in JS can accept and returns other functions

- **Recursion**
  - In a pure FP language there is no iteration! No for loops, while etc.
  - Iteration is achieved using recursion

FP WITH ARRAYS IN JAVASCRIPT

- The Array class in JS implements array instance methods that are higher-order functions.
  - forEach()
  - every()
  - some()
  - find()
  - filter()
  - map()
  - reduce()
- These serve as reusable building blocks to solve a wide variety of problems
- Chaining helps use these building blocks to conveniently piece together a solution
- Helps code become shorter, self-descriptive, and more durable

CUSTOM FP-STYLE ARRAY METHODS

- It is not difficult to implement these built-in methods
  - Example: filter()

- Let's add some custom methods and use them
  - concatAll() – flatten an array with subarrays (arrays as items)
  - concatMap() – calls map() to project and then concatAll() to flatten
  - Zip() – combine items from 2 separate arrays

LIBRARIES WITH FP-STYLE UTILITIES

- Ramda - http://ramdajs.com/
- Lodash - https://lodash.com/