
Functional Programming
(what it is, why do you care)

Composition is KING

Functional programming means coding by composing functions. In other words, programs can be constructed through the definition and composition of functions

Code is DATA

**Functional programming means that functions are first-class citizens of the programming language,
just like regular values.
I can build a list of functions, I can pass them as arguments to other functions, they can be the results returned by other functions**

Expression-Oriented Programming

Application code that is normally written as loops, if-statements and juggling-with-array-indexes becomes infinitely more readable / maintainable when it's rewritten in terms of MAP, REDUCE, FILTER, ZIP and ternary operators

Equational Reasoning

Functional programming performs computations the way mathematicians solve equations - by avoiding global mutable state.
The equal sign does not represent assignment, it represents a binding between a certain identity and a certain value, or an equation

Lazy Evaluation

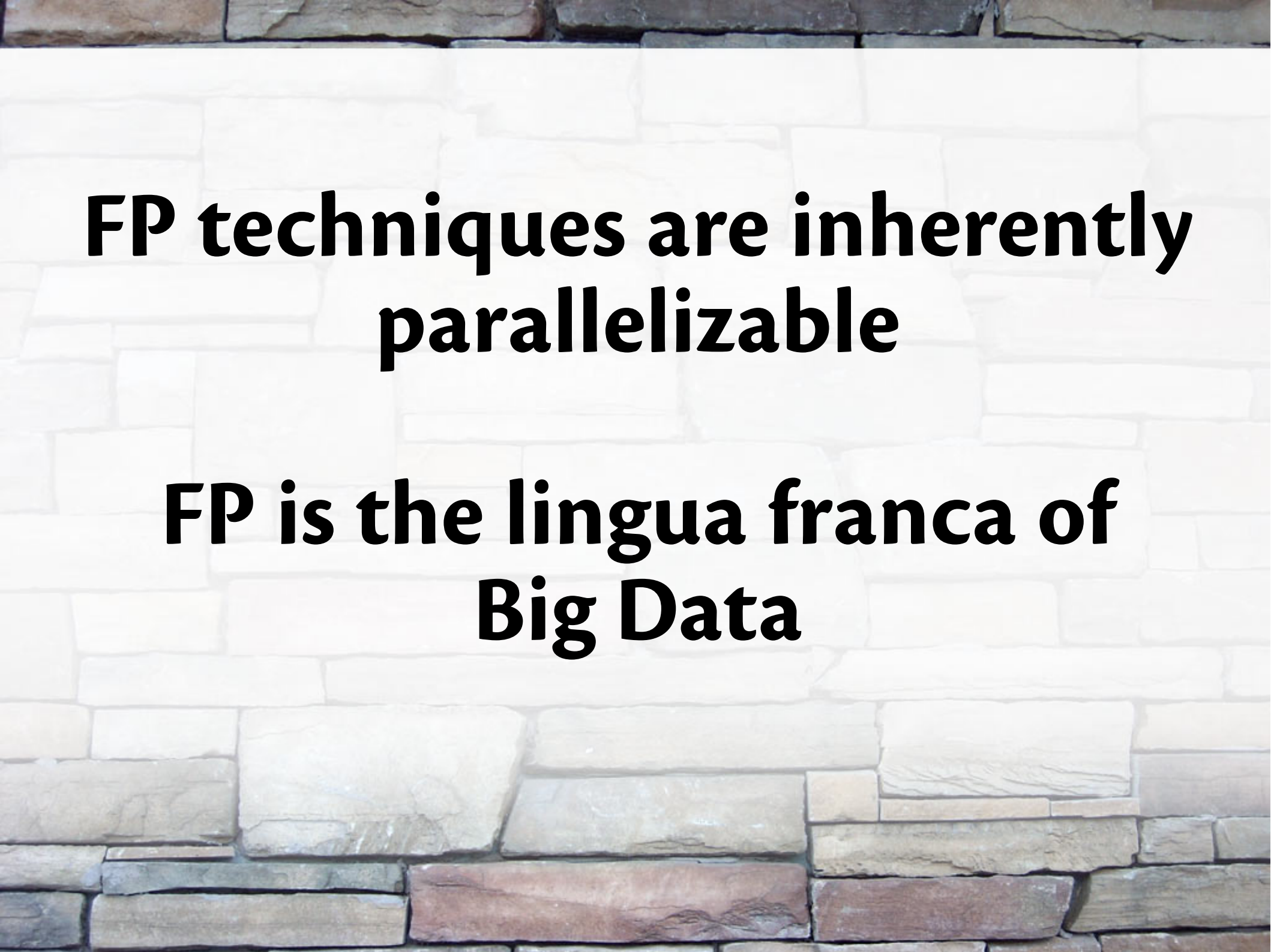
Application logic that deals with infinite streams of data (like infinite scrolling in web applications) becomes more readable and will run in constant memory by leveraging lazy evaluation, a staple of functional programming

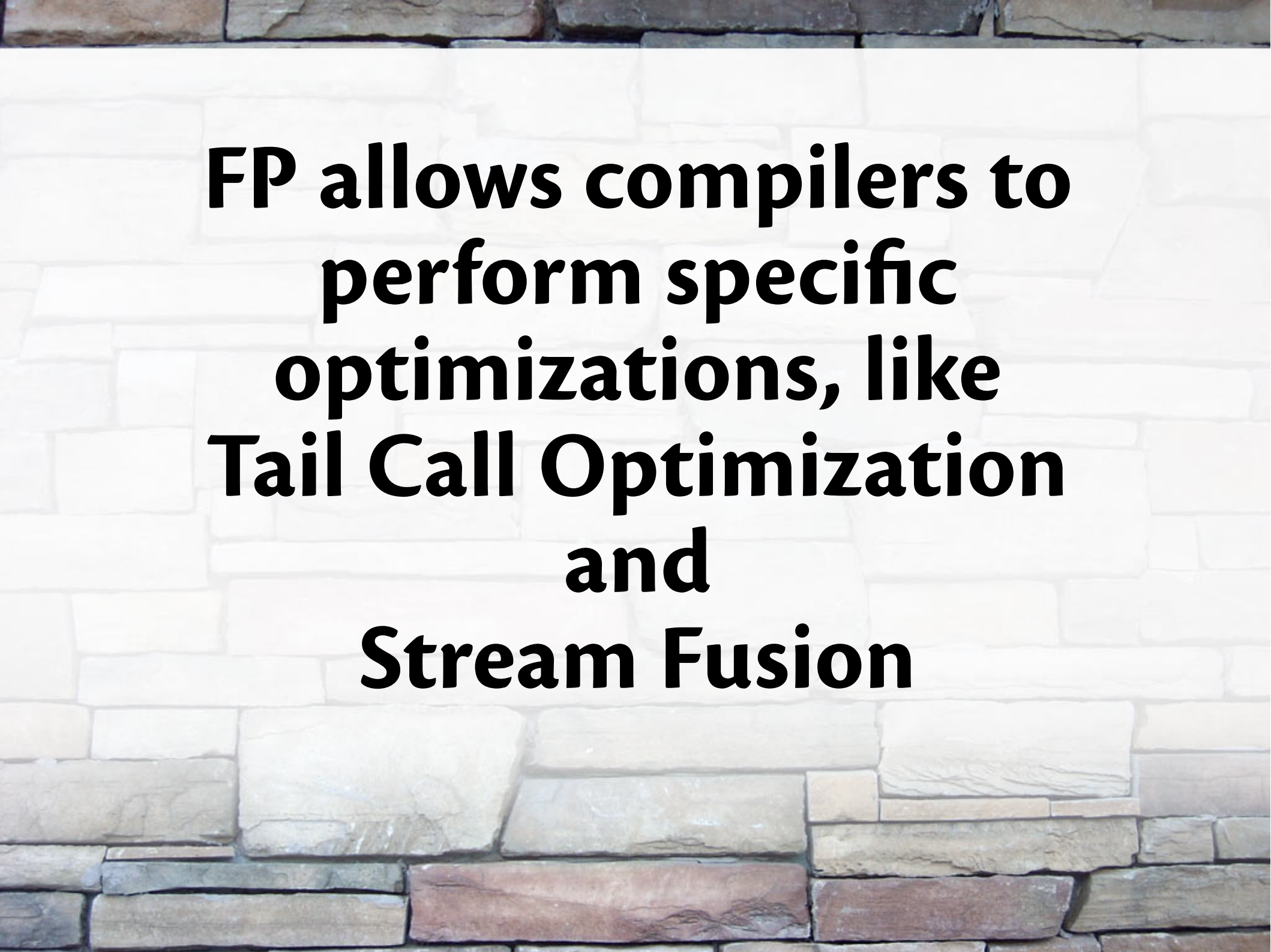
WHY?

Why Functional Programming?

Why Do You Care?


Managing state
in a multi-core
hardware environment
is a nightmare
FP makes it easy


FP techniques are inherently parallelizable
FP is the lingua franca of Big Data


FP allows compilers to perform specific optimizations, like Tail Call Optimization and Stream Fusion


FP provides high-level
abstractions,
making the code
more powerful

FP enables programming by pattern matching

Pattern matching means treating the constituents of a data structure as constituents of a pattern. It makes manipulating complex data structures convenient and expressive


Follow me on GitHub
github.com/dserban