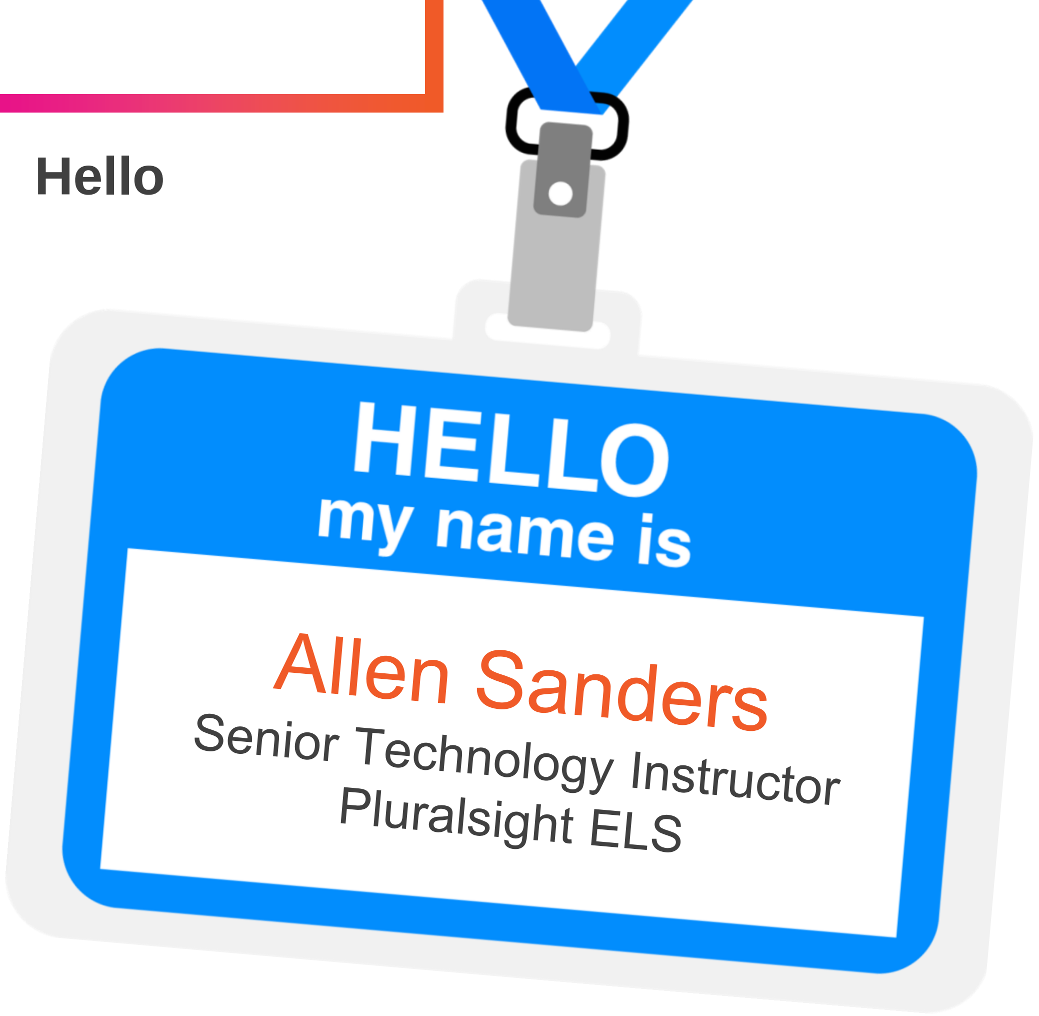
Welcome

Lambdas & Streams in Java

PLURALSIGHT

Hello



About me...

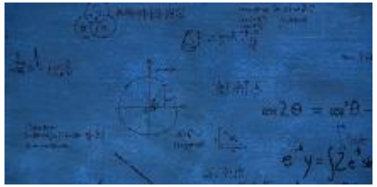
- 27+ years in the industry
- 23+ years in teaching
- Certified Cloud architect
- Passionate about learning
- Also, passionate about Reese's Cups!

Agenda

- Functional Interfaces
- Lambdas
- Streams

Functional Programming

What is It?

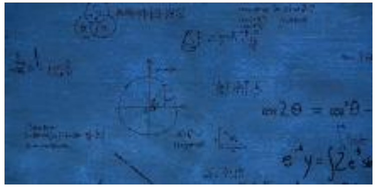


A style of programming where systems are constructed by applying and composing functions that follow a specific set of principles and patterns

Produce same output for same input – no side effects

“Pure”-ness

What is It?



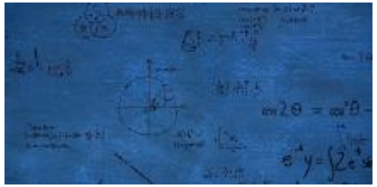
A style of programming where systems are constructed by applying and composing functions that follow a specific set of principles and patterns

Values of variables don’t change – state stays constant

“Pure”-ness

Immutability

What is It?



A style of programming where systems are constructed by applying and composing functions that follow a specific set of principles and patterns

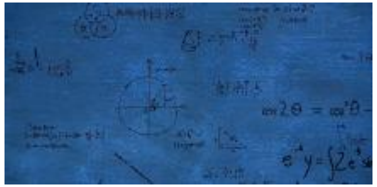
Functions are objects that can be passed as arguments or returned

“Pure”-ness

Immutability

Functions as Objects

What is It?



A style of programming where systems are constructed by applying and composing functions that follow a specific set of principles and patterns

“Pure”-ness

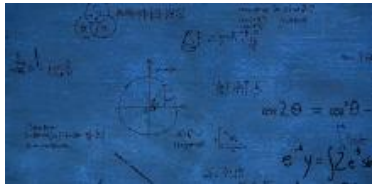
Immutability

Functions as Objects

Recursive

Iterate using recursion instead of looping – implemented using call composition

What is It?



A style of programming where systems are constructed by applying and composing functions that follow a specific set of principles and patterns

“Pure”-ness

Immutability

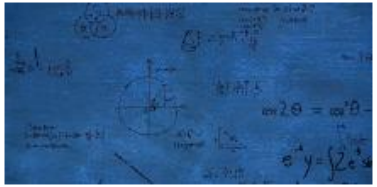
Functions as Objects

Recursive

Transparently Replaceable

Expressions replaceable with values – no side effects

Advantages

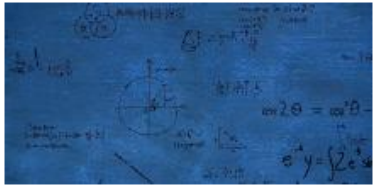


Easier to Debug & Test

Facilitates Concurrency/Parallelism

Readability & Maintainability

Advantages



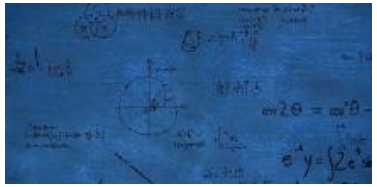
"Pure"-ness and immutability – functions depend only on input arguments and no side effects

Easier to Debug & Test

Facilitates Concurrency/Parallelism

Readability & Maintainability

Advantages



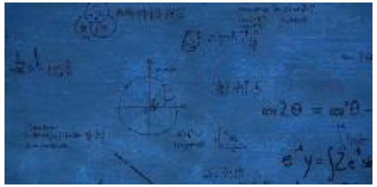
Easier to Debug & Test

Facilitates Concurrency/Parallelism

Because no change to external data or variables, easier to parallelize

Readability & Maintainability

Advantages



Easier to Debug & Test

Facilitates Concurrency/Parallelism

Readability & Maintainability

Goal is small, modular functions – easier to read, understand, and maintain

Functional Interfaces

Functional Interfaces

- Introduced in Java 8 to support Lambdas
- Any Java interface that contains one and only one abstract method
- Rich preexisting set defined in the java.util.function package - https://docs.oracle.com/javase/8/docs/api/java/util/function/package-summary.html

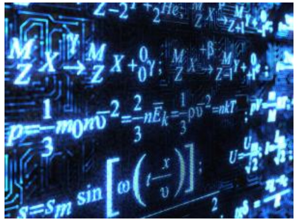
Functional Interfaces

Includes 3 general types (among others):

- Function – Used for Lambda that maps object of one type to object of another type
- Consumer – Used for Lambda that iterates over many objects
- Predicate – Used for Lambda that filters a collection

Lambdas

Lambda Syntax



- Method which implements the single method defined in a functional interface
- (param list) -> { <code block> }
- No method name, sometimes called anonymous function
- Context used to determine parameter types and return type
- If body contains single statement, then curly brackets and return keyword can be omitted
- If single parameter, parentheses not required; if no parameters or multiple parameters, parentheses are required

Streams

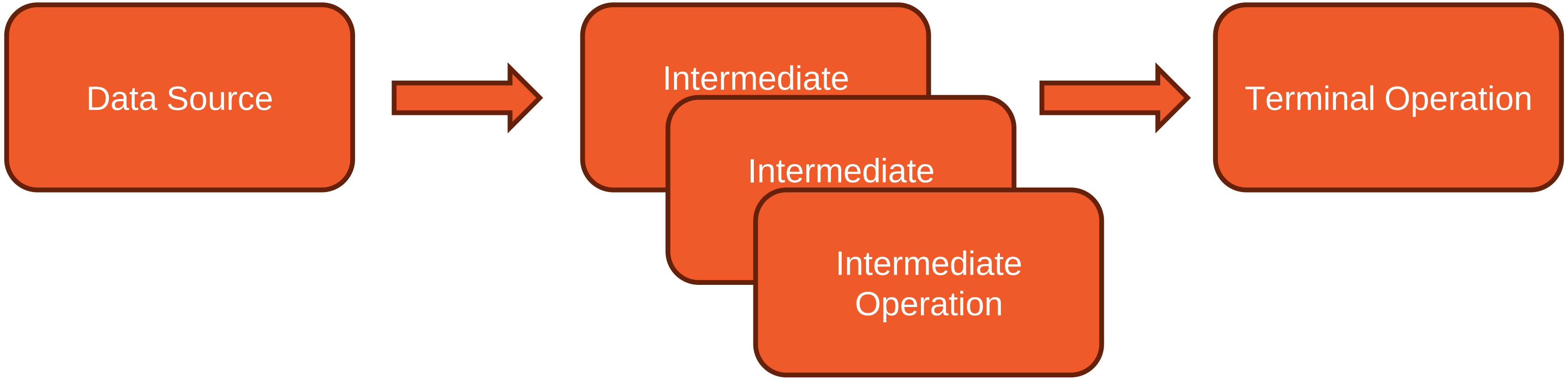
Streams

- Uses elements of Java language to optimize processing of collections and arrays compared to simple iteration

- Especially useful for large collections where each element includes significant processing logic (sometimes called the N*Q profile)

- Multiple ways to create including using the stream() method, Stream.of(), Stream.iterate(), Stream.generate(), Stream.empty(), etc.

Streams


Data Source
Intermediate
Intermediate
Intermediate Operation
Terminal Operation

Thank you!

If you have additional questions, please reach out to me at: asanders@gamuttechnologysvcs.com

PLURALSIGHT