Understanding Collections

Harit Himanshu
@harittweets

Overview

Understanding how `reduce` works

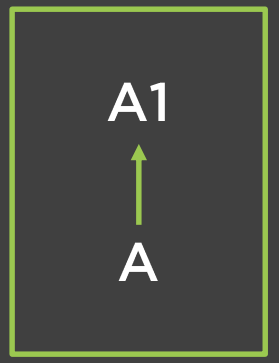
Understanding how `fold` works

Project Demo

What Does Reduce Do?

Reduce Signature

```
def reduce[A1 >: A](op: (A1, A1) ⇒ A1): A1
```



op(String, String) ⇒ String

op(Int, Int) ⇒ Int

op(Dollar, Dollar) ⇒ Dollar

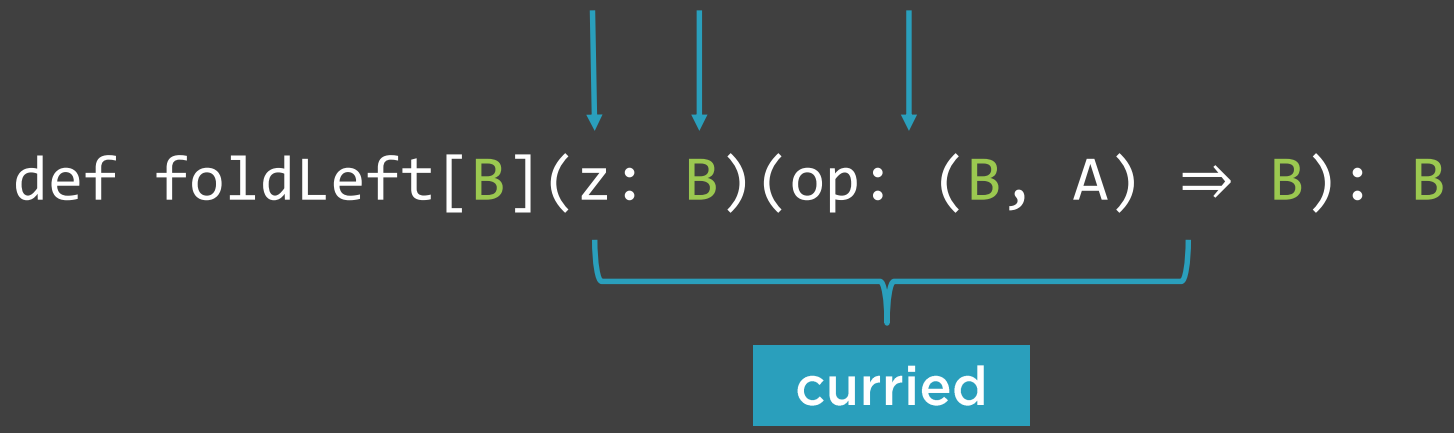
op(Account, Account) ⇒ Account

What Is Operator Doing?

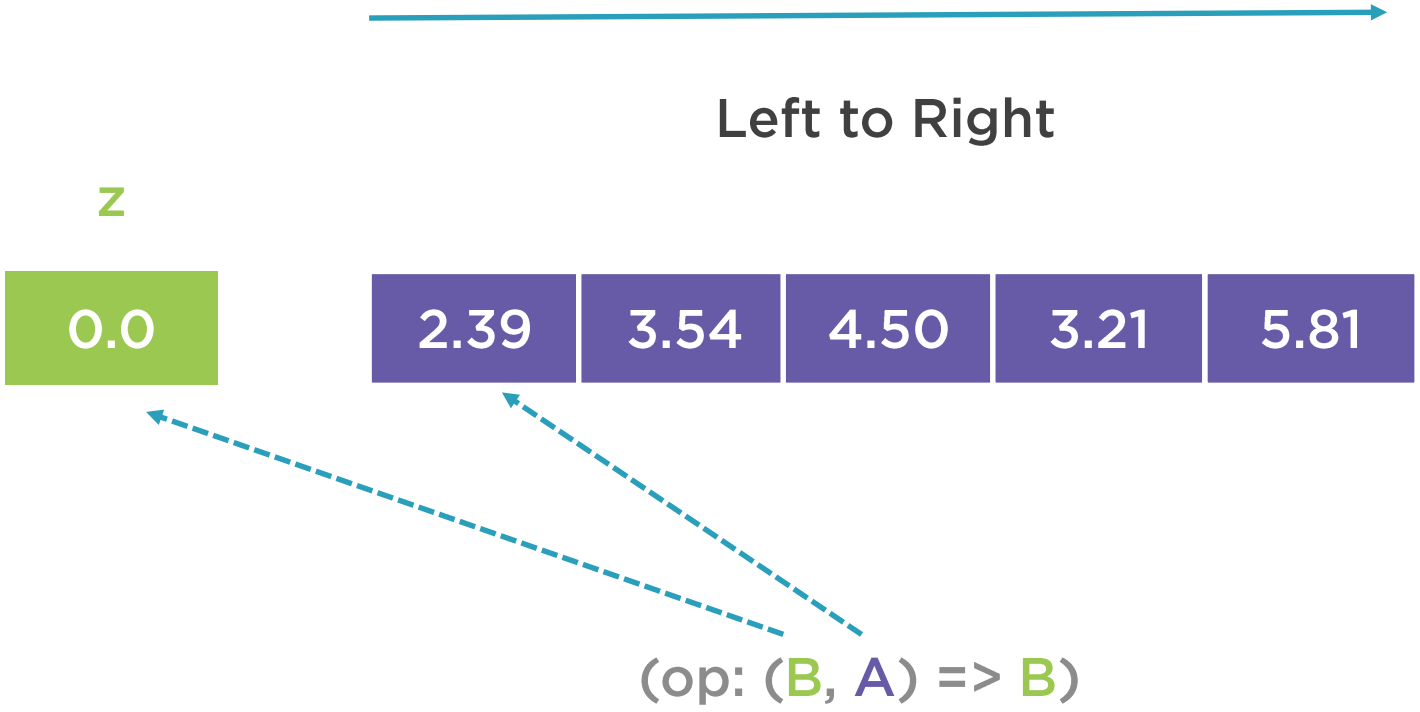
Operators? +, -, min, max, sum, multiply

associative

FoldLeft Signature



How Does FoldLeft Work?

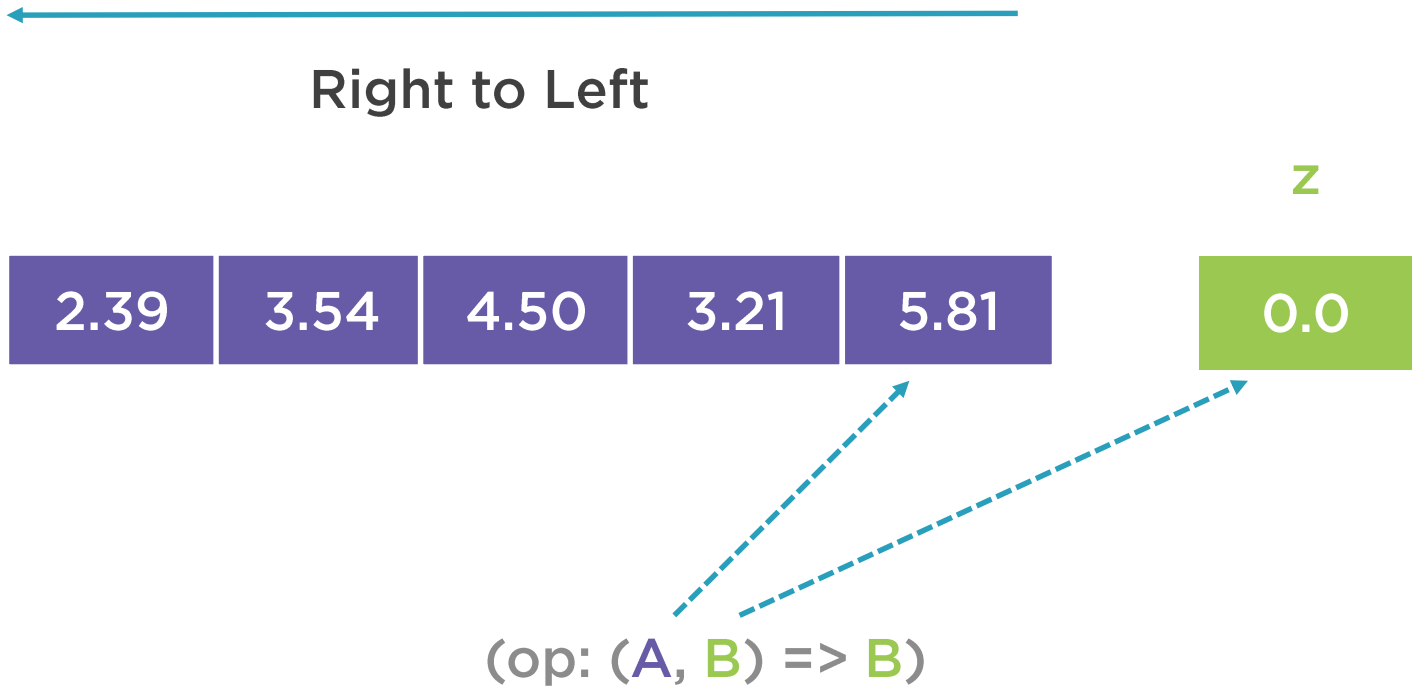


FoldRight Signature

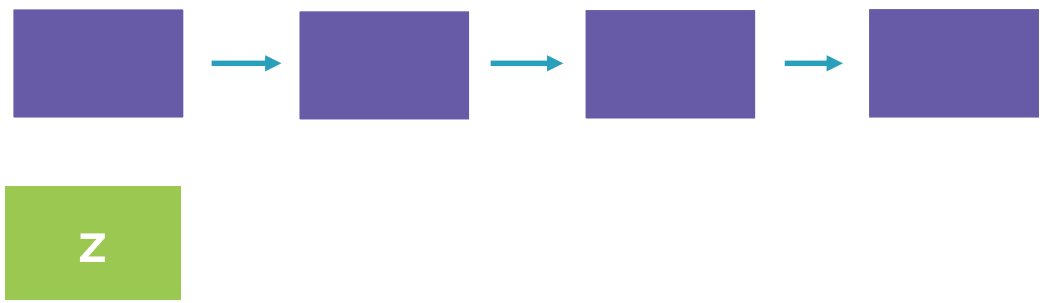
```
def foldRight[B](z: B)(op: (A, B) ⇒ B): B

def foldLeft[B](z: B)(op: (B, A) ⇒ B): B
```

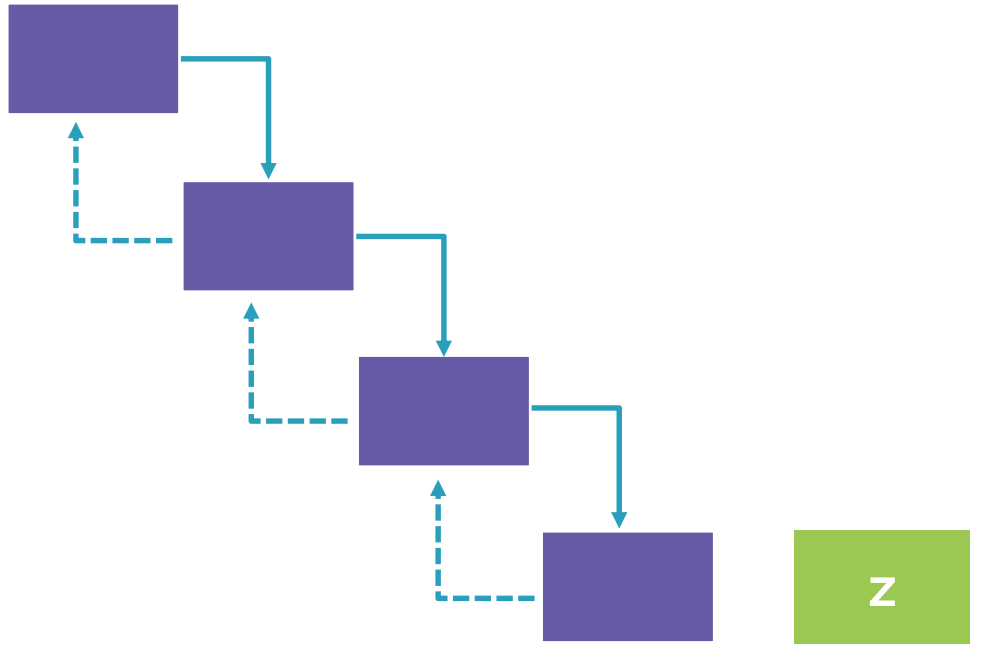
How Does FoldRight Work?



FoldLeft vs. FoldRight Computation Model



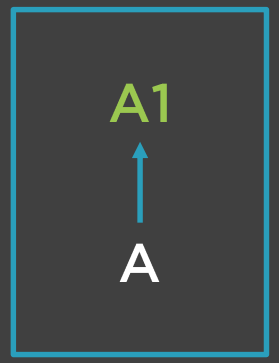
FoldLeft



FoldRight

Fold Signature

```
def fold[A1 >: A](z: A1)(op: (A1, A1) ⇒ A1): A1
```



Nil => List concatenation

0 => addition

1 => product

Which Fold Do I Use?

FoldLeft

iterative

deterministic

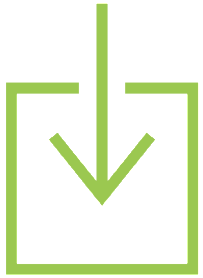
Fold

FoldRight

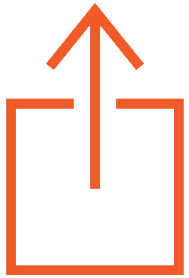
Demo

- Refactor the `Product` Class
- Refactor the `Account` Class
- Create a new `StatisticsService` Trait
- Refactor the `BankOfScala` Class

Bank Growth Metrics



Money Deposited



Money Borrowed

Account Activity

Summary

Understanding how `reduce` works

Understanding how `fold` works

Project Demo