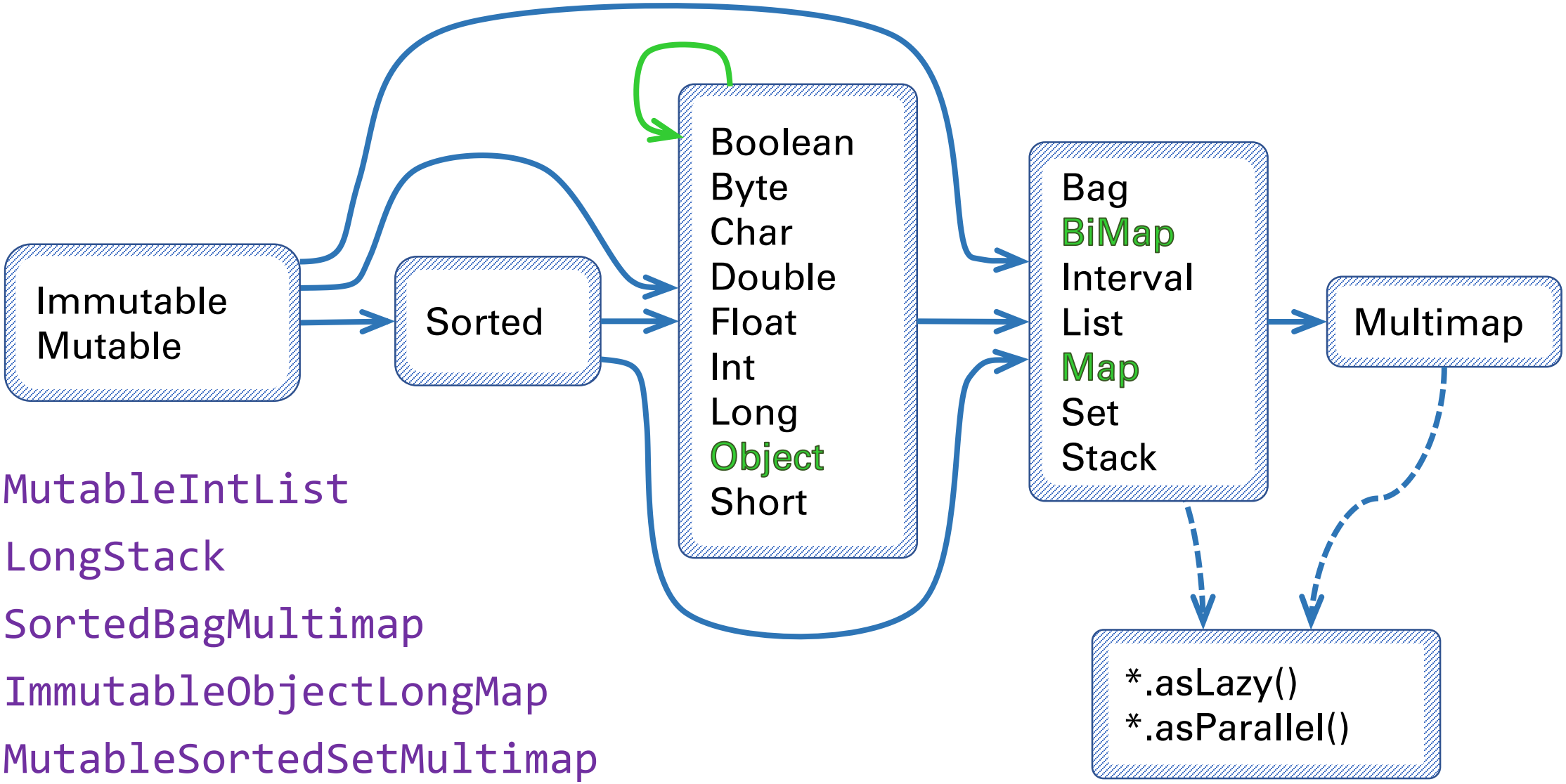
Refactoring to Eclipse Collections

Making Your Java Streams Leaner, Meaner, and Cleaner

Introduction

- What is Eclipse Collections?
  - Feature rich, memory efficient Java Collections framework
- History
  - Eclipse Collections started off as an internal collections framework named Caramel at Goldman Sachs in 2004
  - In 2012, it was open sourced to GitHub as a project called GS Collections
  - GS Collections was migrated to the Eclipse Foundation, re-branded as Eclipse Collections in 2015
- Eclipse Collections open for contributions!

Build Any Types You Need




```
MutableIntList
LongStack
SortedBagMultimap
ImmutableObjectLongMap
MutableSortedSetMultimap
```

Methods [some of] by Category

transform

collect
collectBoolean
collectByte
collectChar
collectDouble
collectFloat
collectIf
collectInt
collectKeysAndValues
collectLong
collectShort
collectValues
collectWith
collectWithIndex
collectWithOccurrences
flatCollect

wrap

asLazy
asParallel
asReversed
asSynchronized
asUnmodifiable

group

groupBy
groupByEach
groupByUniqueKey
sumByDouble
sumByFloat
sumByInt
sumByLong
aggregateBy
aggregateInPlaceBy

convert

toArray
toBag
toImmutable
toIntArray
toList
toMap
toMapOfItemToCount
toReverseArray
toReverseList
toReversed
toSet
toSortedBag**[By]**
toSortedList**[By]**
toSortedMap
toSortedSet
toSortedSet**[By]**
toStack
toString
toStringOfItemToCount

filter

select**[With]**
selectByOccurrences
selectInstancesOf
reject**[With]**
partition**[With]**
partitionWhile

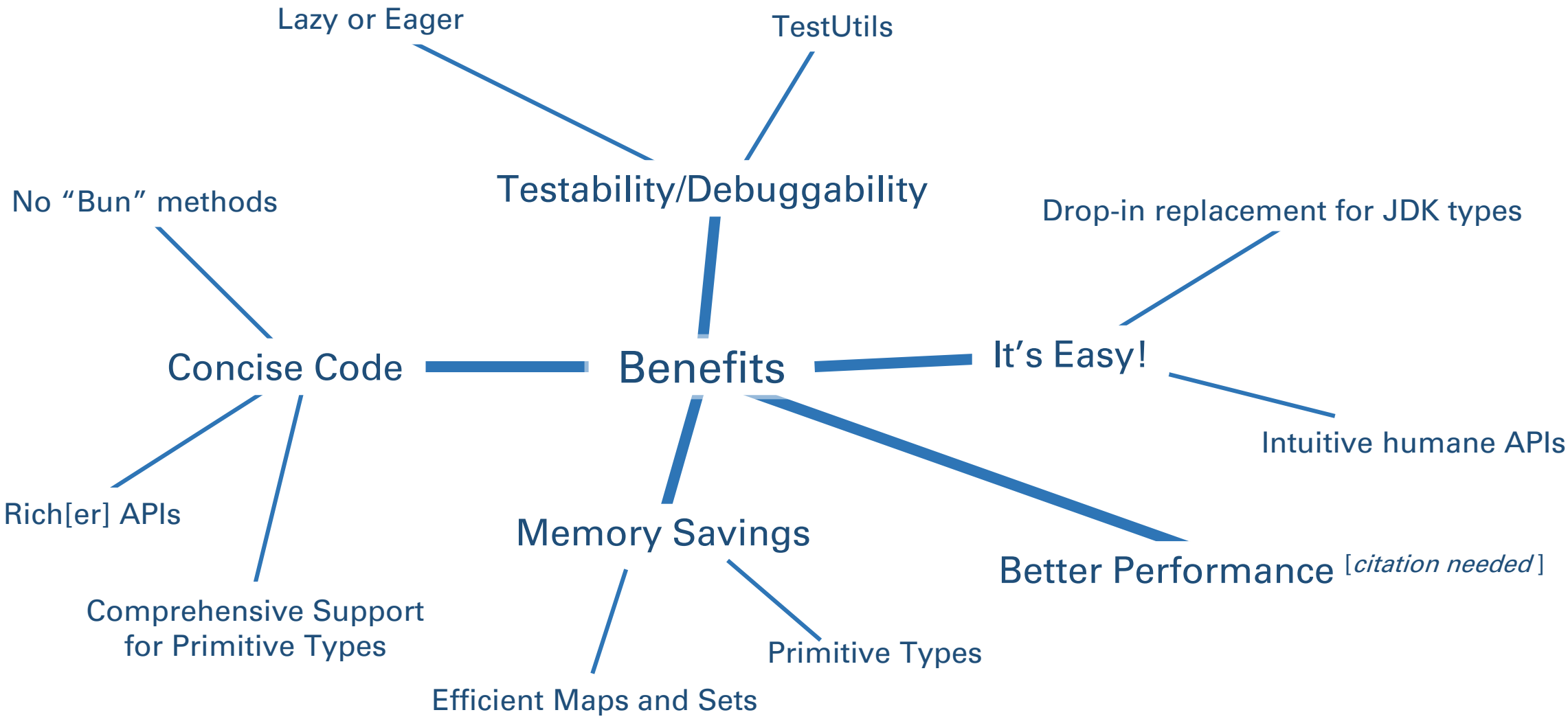
test

allSatisfy**[With]**
anySatisfy**[With]**
noneSatisfy**[With]**
notEmpty
isEmpty

Methods – lots more…

Word sizes are proportional to the number of implementations of the method

Why Refactor to EC?



Let’s Do It!

JMH Benchmark Results

Memory Usage Comparison