ORACLE

Lesson 2-3: Streams of Objects and Primitive Types

Objects And Primitives

Summary

- The Java language is not truly object oriented
- Primitive types are included
  - `byte`, `short`, `int`, `long`, `double`, `float`, `char`
- For some situations these are wrapped as objects
  - E.g. storage in collections
  - `Byte`, `Short`, `Integer`, etc.
- Conversion between primitive and object representation is often handled by auto-boxing and unboxing

Object Streams

Slides 3 & 4 aren't quite complete. Getting this code to compile is explained in Lesson 2-7.

- By default, a stream produces elements that are objects
- Sometimes, this is not the best solution

```
int highScore = students.stream()
   .filter(s -> s.graduationYear() == 2015)
   .map(s -> s.getScore())
   .max(Integer::compare);
```

The stream from map has to auto-box ints to objects

`max()` must unbox each Integer object to get the value

`getScore()` returns a primitive int

Primitive Streams

- To avoid a lot of unnecessary object creation and work we have three primitive stream types
  - `IntStream`, `DoubleStream`, `LongStream`
- These can be used with certain stream operations

```
int highScore = students.stream()
   .filter(s -> s.graduationYear() == 2015)
   .mapToInt(s -> s.getScore())
   .max();
```

The stream from `mapToInt` is a stream of int values, so no boxing or unboxing

Section 3

Summary

- Java has primitive values as well as object types
- To improve stream efficiency we have three primitive stream types
  - `IntStream`, `DoubleStream`, `LongStream`
- Use methods like `mapToInt()`, `mapToDouble()`, `mapToLong()`


ORACLE®