CompletionService Interface

This lesson talks about how to batch multiple tasks together

CompletionService Interface

In the previous lesson we discussed how tasks can be submitted to executors but imagine a scenario where you want to submit hundreds or thousands of tasks. You'll retrieve the future objects returned from the submit calls and then poll all of them in a loop to check which one is done and then take appropriate action. Java offers a better way to address this use case through the `CompletionService` interface. You can use the `ExecutorCompletionService` as a concrete implementation of the interface.

The completion service is a combination of a blocking queue and an executor. Tasks are submitted to the queue and then the queue can be polled for completed tasks. The service exposes two methods, one `poll` which returns null if no task is completed or none were submitted and two `take` which blocks till a completed task is available.

Below is an example program that demonstrates the use of completion service.

```
1  import java.util.Random;
2  import java.util.concurrent.Ex
3  import java.util.concurrent.Ex
4  import java.util.concurrent.Ex
5  import java.util.concurrent.Fu
6
7
8  class Demonstration {
9
10   static Random random = new R
11
12   public static void main( Str
13       completionServiceExampl
14   }
15
```

```
16
17  static void completionServic
18
19      class TrivialTask impl
20
21          int n;
22
23          public TrivialTask
24              this.n = n;
25          }
26
27          public void run() 
28              try {
29                  // sleep f
30                  Thread.sle
31                  System.out
```