Review of the Basics

Spark DStreams use abstractions called RDDs

- Incoming data is divided into chunks that can be operated on in parallel or in series
  - Remember our Coffee house analogy
- Operations on one stream or set of RDDs results in a new stream or set of RDDs
  - The result is a growing chain of dependencies
- In some cases, we can use Spark SQL to make DStreams behave like SQL dataframes.

Can use a variety of functions on DStreams

Some examples of what we've covered include:

- Lambda functions
  - Functional programming is extremely useful for operations on streams
- Transformations
  - Either built in (map, flatmap, union, groupBy, etc.), or custom-made using the Transform() operation.
- Window functions
  - window(), countByWindow(), reduceByKeyAndWindow(), countByValueAndWindow() work on intervals of time in your stream

DStreams have a variety of input & output options

- Can take in DStream from a source
  - ssc.textFileStream(dataDirectory)
  - ssc.socketTextStream(host, port)
- Can create a DStream from scratch
  - ssc.queueStream(queueOfRDDs)
  - ssc.queueStream(parallelize([arrayOfRDDInputs]))
- Can output to a file or console
  - ssc.saveAsTextFiles(prefix, [suffix])
  - pprint()
  - ssc.foreachRDD(customFunction)

To the Final Exercise!