Spark-Streaming:

**Data** **Access**:

* Spark streaming provides a high-level abstraction called discretized stream or DStream.
* Sequence of RDD forms a DStream.
* Each RDD in DStream contains data from a certain interval.

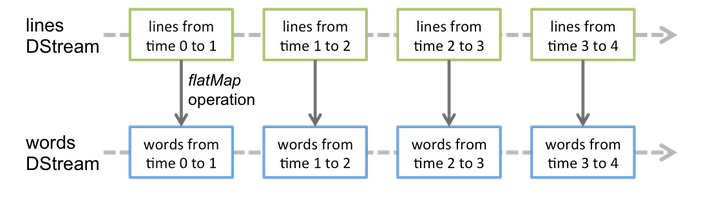


* Every input Dstream receive a single stream of data.
* We can have multiple input Dstream.



**Operations:**

* Operations applied on Dstream is translated to underlying RDD



**Streaming Sources:**

* There are two types of Streaming sources available.
  + Basic Sources : Sources which are directly available in StreamingContext API

File systems, socket connections, Akka actors.

* + Advance Sources: Sources which are available through extra utility classes. (required linking against extra dependencies)

Kafka, Flume, Kinesis, Twitter

**Background Process:**

* Receiver object designed in java.
* Input Dstream is associated with a single Receiver object (expect file system)
* Receiver receives the data from the source and store in spark memory for processing.
* Receiver runs within spark as a long-running task.
* It occupies one of the cores allocated to the spark streaming application.
* Hence, we must have enough cores to process the received data.

**NOTE:-**

If the number of cores allocated to the application is less than or equal to the number of input Dstream/receivers, then the system will receive data, but not be able to process them. Must have more than one core to process the data, as the one core is occupied by the receiver.

**File Streaming:**

* To use files for the streaming we have to use streamingContext.fileStream library.
* streamingContext.fileStream[keyClass, valueClass, inputFormatClass](dataDirectory)
* We can access any file which is on any filesystem which is compatible with HDFS and to create a Dstream
* Files created in nested directories are not processed.

**NOTE:-**

* Files must have same data format.
* Once placed the file in dataDirectory must not be changed. The appended data will not be read.
* For simple text files spark streaming provides a method streamingContext.textFileStream(dataDirectory) which don’t required a running receiver and hence no core is required.

**Other Methods:-**

* streamingContext.actorStream(actorProps, actor-name): To create Dstreams using data received through AKKA actors.
* streamingContext.queueStream(queueOfRDDs): To create Dstream based on queue of RDD’s. Each RDD is pushed into the queue will be treated as a batch of data in the Dstream.

**Transformations on DStreams**

* All the transformations which we have for RDD exists and work on same way to Dstreams.
* Apart from that we have a new Transformations for Dstreams.
  + **UpdateStateByKey**