스파크 스트리밍 응용 프로그램

1 스파크 스트리밍 응용 프로그램 작성 단계

① TCP 소켓을 통해 텍스트 줄을 받고 각 단어가 나타나는 횟수를 계산하는 프로그램 예시

응용 프로그램 작성 단계

1. StreamingContext 만들기
2. StreamingContext에서 Dstream 만들기
3. Dstream에 변환을 적용하기
4. 결과 출력하기

스파크 스트리밍 응용 프로그램

2 Scala 스파크 스트리밍 응용 프로그램 작성하기

① StreamingContext 만들기

SparkContext에서 StreamingContext를 생성

클러스터

StreamingContext를 만들 때 초로 일괄 처리의 크기를 지정

```
Import org.apache.spark._
import org.apache.spark.streaming._
import org.apache.spark.streaming.StreamingContext._ // not necessary since Spark 1.3

// create a local StreamingContext with two working thread and batch interval of 1 second.
// The master requires 2 cores to prevent from a starvation scenario.

val conf = new SparkConf().setMaster("local[2]").setAppName("NetworkWordCount")
val ssc = new StreamingContext(conf, Seconds(1))
```

스파크 스트리밍 응용 프로그램

2 Scala 스파크 스트리밍 응용 프로그램 작성하기

② StreamingContext에서 DStream 만들기

StreamingContext 인스턴스를 사용 → 입력 원본에 대한 입력 Dstream을 생성함

```
// create a DStream that will connect to hostname:port, like localhost:9999
val lines = ssc.socketTextStream("localhost", 9999)
```

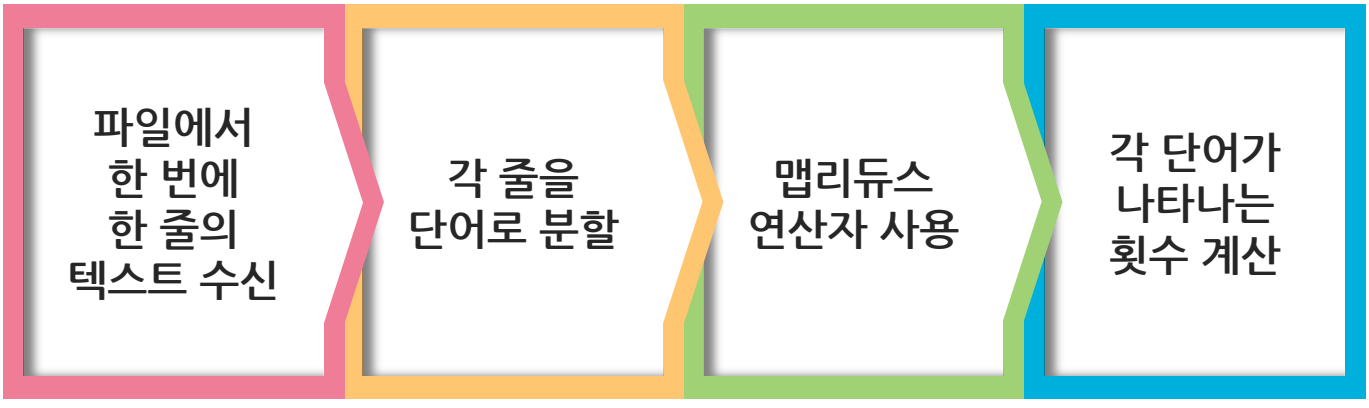
③ DStream에 변환을 적용하기

스파크 스트리밍 응용 프로그램

2 Scala 스파크 스트리밍 응용 프로그램 작성하기

3 DStream에 변환을 적용하기

응용 프로그램




```
// Split each line into words
val words = lines.flatMap(_.split(" "))

import org.apache.spark.streaming.StreamingContext._ // not necessary since Spark 1.3
// Count each word in each batch
val pairs = words.map(word => (word, 1))
val wordCounts = pairs.reduceByKey(_ + _)
```

스파크 스트리밍 응용 프로그램

2 Scala 스파크 스트리밍 응용 프로그램 작성하기

4 결과 출력하기

출력 작업을 적용하여 대상 시스템에 변환 결과를 푸시함

콘솔 출력에서 계산을 통해 각 실행 결과를 표시

```
// Print the first ten elements of each RDD generated in this DStream to the console
wordCounts.print()
```

5 스트리밍 응용 프로그램 시작부터 종료 신호 수신까지 실행

```
ssc.start()             // Start the computation
ssc.awaitTermination()  // Wait for the computation to terminate
```

스파크 스트리밍 응용 프로그램

3 Java 스파크 스트리밍 응용 프로그램 작성하기

1 StreamingContext 만들기

SparkContext에서 StreamingContext를 생성

클러스터

StreamingContext를 만들 때 초로 일괄 처리의 크기를 지정

```
import org.apache.spark.*;
import org.apache.spark.api.java.function.*;
import org.apache.spark.streaming.*;
import org.apache.spark.streaming.api.java.*;
import scala.Tuple2;

// create a local StreamingContext with two working thread and batch interval of 1 second

SparkConf conf = new SparkConf().setMaster("local[2]").setAppName("NetworkWordCount");
JavaStreamingContext jssc = new JavaStreamingContext(conf, Durations.Seconds(1));
```

스파크 스트리밍 응용 프로그램

3 Java 스파크 스트리밍 응용 프로그램 작성하기

2 StreamingContext에서 DStream 만들기

StreamingContext 인스턴스를 사용 → 입력 원본에 대한 입력 Dstream을 생성함

```
// create a DStream that will connect to hostname:port, like localhost:9999
JavaReceiverInputDStream<String> lines = jssc.socketTextStream("localhost", 9999);
```

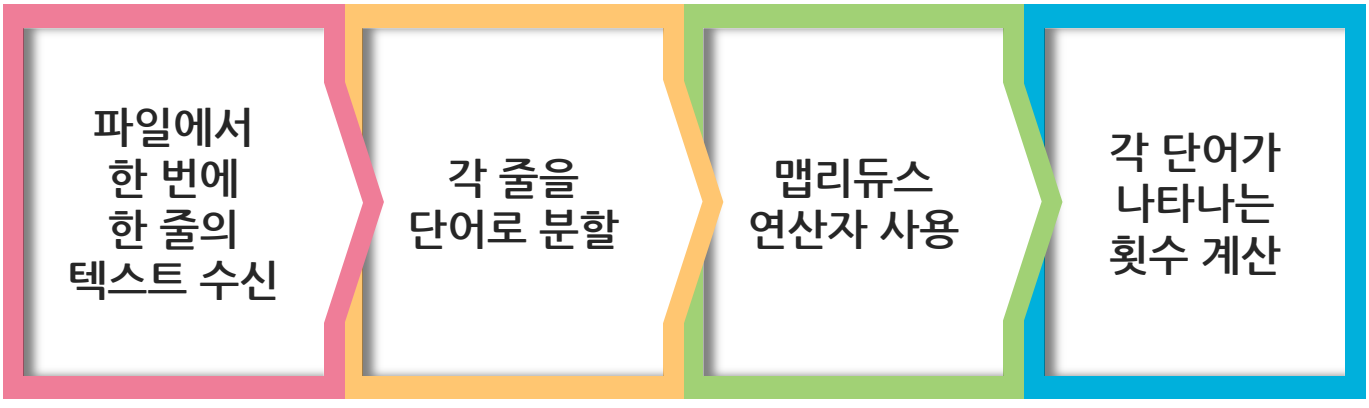
3 DStream에 변환을 적용하기

스파크 스트리밍 응용 프로그램

3 Java 스파크 스트리밍 응용 프로그램 작성하기

3 DStream에 변환을 적용하기

응용 프로그램




```
// Split each line into words
JavaDStream<String> words = lines.flatMap(x -> Arrays.asList(x.split(" ")).iterator());
```

스파크 스트리밍 응용 프로그램

3 Java 스파크 스트리밍 응용 프로그램 작성하기

4 결과 출력하기

출력 작업을 적용하여 대상 시스템에 변환 결과를 푸시함

콘솔 출력에서 계산을 통해 각 실행 결과를 표시

```
// Count each word in each batch
JavaPairDStream<String, Integer> Pairs = words.mapToPair(s -> new Tuple2<>(s, 1));
JavaPairDStream<String, Integer> wordCounts = pairs.reduceByKey((i1, i2) -> i1 + i2);

// Print the first ten elements of each RDD generated in this DStream to the console
wordCounts.print()
```

5 스트리밍 응용 프로그램 시작부터 종료 신호 수신까지 실행

```
jssc.start()              // Start the computation
jssc.awaitTermination()   // Wait for the computation to terminate
```

스파크 스트리밍 응용 프로그램

4 Python 스파크 스트리밍 응용 프로그램 작성하기

1 StreamingContext 만들기

SparkContext에서 StreamingContext를 생성

클러스터

StreamingContext를 만들 때 초로 일괄 처리의 크기를 지정

```
from pyspark import SparkContext
from pyspark.streaming import StreamingContext

# Create a local StreamingContext with two working thread and batch interval of 1 second
sc = SparkContext("local[2]", "NetworkWordCount")
ssc = StreamingContext(sc, 1)
```

스파크 스트리밍 응용 프로그램

4 Python 스파크 스트리밍 응용 프로그램 작성하기

② StreamingContext에서 DStream 만들기

StreamingContext 인스턴스를 사용 → 입력 원본에 대한 입력 Dstream을 생성함

```
# Create a DStream that will connect to hostname:port, like localhost:9999
lines = ssc.socketTextStream("localhost", 9999)
```

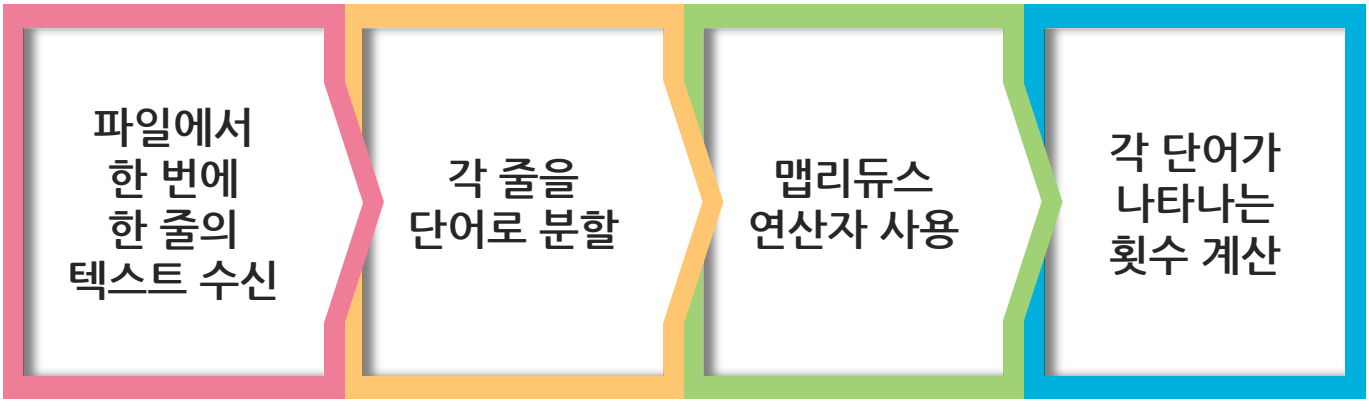
③ DStream에 변환을 적용하기

스파크 스트리밍 응용 프로그램

4 Python 스파크 스트리밍 응용 프로그램 작성하기

3 DStream에 변환을 적용하기

응용 프로그램




```
# Split each line into words
words = lines.flatMap(lambda line: line.split(" "))
```

스파크 스트리밍 응용 프로그램

4 Python 스파크 스트리밍 응용 프로그램 작성하기

④ 결과 출력하기

출력 작업을 적용하여 대상 시스템에 변환 결과를 푸시함

콘솔 출력에서 계산을 통해 각 실행 결과를 표시

```
# Count each word in each batch
Pairs = words.map(lambda word: (word, 1))
wordCounts = pairs.reduceByKey(lambda x, y: x + y)

// Print the first ten elements of each RDD generated in this DStream to the console
wordCounts.pprint()
```

⑤ 스트리밍 응용 프로그램 시작부터 종료 신호 수신까지 실행

```
ssc.start()                    // Start the computation
ssc.awaitTermination()    // Wait for the computation to terminate
```

스파크 스트리밍 응용 프로그램

5 스파크 스트리밍 응용 프로그램 실행하기

1 Scala

✓ 데이터 서버로 Netcat을 실행하고 NetworkWordCount 실행하기

```
# TERMINAL 1:
# Running Netcat

$ nc -lk 9999

hello world


...
```

```
# TERMINAL 2: RUNNING NetworkWordCount

$ ./bin/run-example streaming.NetworkWordCount localhost 9999
...
-------------------------------------------
Time: 1357008430000 ms
-------------------------------------------
(hello, 1)
(world, 1)
...
```

2 Java

✓ 데이터 서버로 Netcat을 실행하고 NetworkWordCount 실행하기

```
# TERMINAL 1:
# Running Netcat

$ nc -lk 9999

hello world


...
```

```
# TERMINAL 2: RUNNING JavaNetworkWordCount

$ ./bin/run-example streaming.JavaNetworkWordCount localhost 9999
...
-------------------------------------------
Time: 1357008430000 ms
-------------------------------------------
(hello, 1)
(world, 1)
...
```

스파크 스트리밍 응용 프로그램

5 스파크 스트리밍 응용 프로그램 실행하기

3 Python

✓ 데이터 서버로 Netcat을 실행하고 NetworkWordCount 실행하기

```
# TERMINAL 1:
# Running Netcat

$ nc -lk 9999

hello world



...
```

```
# TERMINAL 2: RUNNING network_wordcount.py

$ ./bin/spark-submit examples/src/main/python/streaming/
network_wordcount.py localhost 9999
...
-------------------------------------------
Time: 2014-10-14 15:25:21
-------------------------------------------
(hello, 1)
(world, 1)
...
```