

Week 2: Hadoop MapReduce Streaming Applications in Python

Streaming

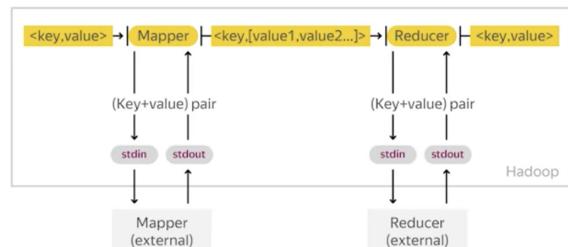
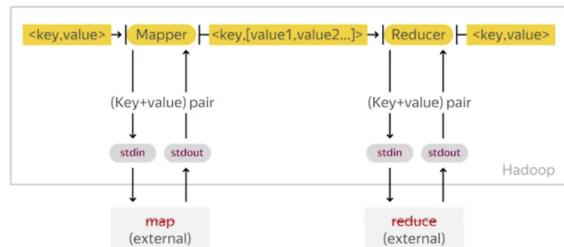
Shell streaming



Scipy와 Numpy같은 C로 만든
라이브러리와 호환이 불가하다.



하둡에서 다양한 언어를 지원



- 표준 입출력을 활용하여 프로그램 코드 작성

Mapper

- define input format
- process data
- define output format

Reducer

- define input format
- aggregate sorted data by key
- process data
- define output format

```
$ man locate //streaming jar 경로 찾기
```

```
HADOOP_STREAMING_JAR="/path/to/hadoop-streaming.jar"
yarn jar $HADOOP_STREAMING_JAR \
  -mapper 'wc -l' \
  -numReduceTasks 0 \
  -input /data/wiki/en_articles \
  -output wc_mr
```

- yarn jar: open yarn service
- -numReduceTasks option: reduce task X

```
HADOOP_STREAMING_JAR="/path/to/hadoop-streaming.jar"
yarn jar $HADOOP_STREAMING_JAR \
  -mapper 'wc -l' \
  -numReduceTasks 0 \
  -input /data/wiki/en_articles \
  -output wc_mr
```

```
ERROR streaming.StreamJob: Error Launching job : Output
directory hdfs://virtual-master.atp-fvt.org:8020/user/adral/
wc_mr already exists
Streaming Command Failed!
```

- output path는 새로워야함
- 작은 데이터에 대해 검증해보는 습관

```
HADOOP_STREAMING_JAR="/path/to/hadoop-streaming.jar"
yarn jar $HADOOP_STREAMING_JAR \
  -mapper 'wc -l' \
  -reducer 'awk '{line_count += \$1} END { print line_count }'" \
  -numReduceTasks 1 \
  -input /data/wiki/en_articles \
  -output wc_mr
```

```
$ hdfs dfs -ls wc_mr
Found 3 items
-rw-r--r-- 3 adral adral 0 2017-03-21 14:48 wc_mr/_SUCCESS
-rw-r--r-- 3 adral adral 6 2017-03-21 14:48 wc_mr/part-00000
-rw-r--r-- 3 adral adral 6 2017-03-21 14:48 wc_mr/part-00001
```

```
$ hdfs dfs -text wc_mr/*
1986
2114
```

1968 + 2114 = 4100

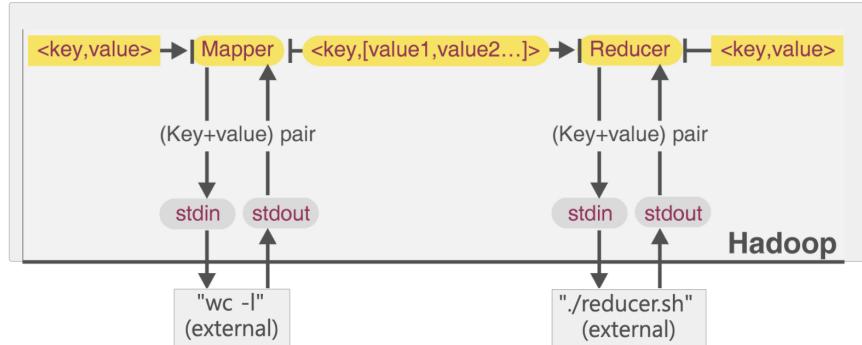
• reducer.sh

sh 파일을 직접 만들어 node에 업로드 후 사용

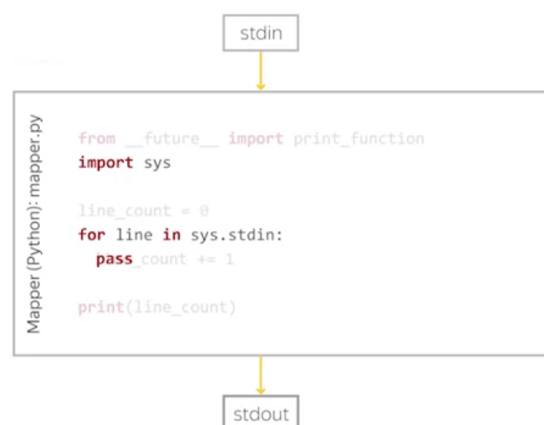
```
#!/usr/bin/env bash
awk '{line_count += $1} END { print line_count }'

HADOOP_STREAMING_JAR = '/path/to/hadoop-streaming.jar'
yarn jar $HADOOP_STREAMING_JAR \
  -mapper 'wc -l' \
  -reducer './reducer.sh' \
```

```
-file reducer.sh \
-numReduceTasks 1 \
-input /data/wiki/en_articles \
-output wc_mr_with_reducer
```



Streaming in Python



```
HADOOP_STREAMING_JAR=/path/to/hadoop-streaming.jar
yarn jar $HADOOP_STREAMING_JAR \
-files mapper.py,reducer.sh \
-mapper 'python mapper.py' \
-reducer './reducer.sh' \
-numReduceTasks 1 \
-input /data/wiki/en_articles \
-output wc_mr_with_reducer
```

The general command line syntax is
bin/hadoop command [genericOptions] [commandOptions]
-conf <configuration file>
-D <property>
-fs <local|namenode:port>
-jt <local|resourcemanager:port>
-files <comma separated list of files>
-libjars <comma separated list of jars>
-archives <comma separated list of archives>

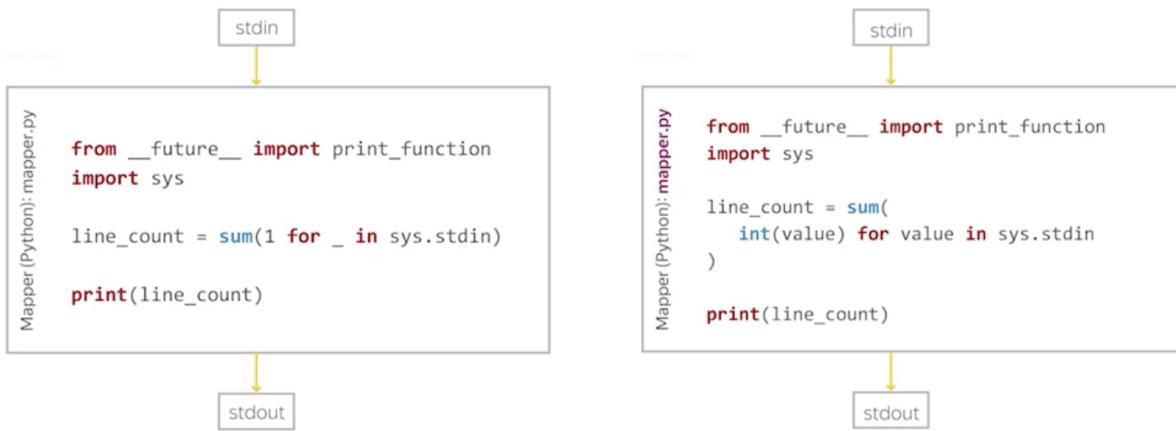
```
HADOOP_STREAMING_JAR=/path/to/hadoop-streaming.jar
yarn jar $HADOOP_STREAMING_JAR \
-files mapper.py,reducer.sh \
-mapper 'python mapper.py' \
-reducer './reducer.sh' \
-numReduceTasks 1 \
-input /data/wiki/en_articles \
-output wc_mr_with_reducer
```

```
$ hdfs dfs -ls wc_mr_with_reducer
Found 2 items
-rw-r--r-- 3 adral adral 0 <date> wc_mr_with_reducer/_SUCCESS
-rw-r--r-- 3 adral adral 0 <date> wc_mr_with_reducer/part-00000
```

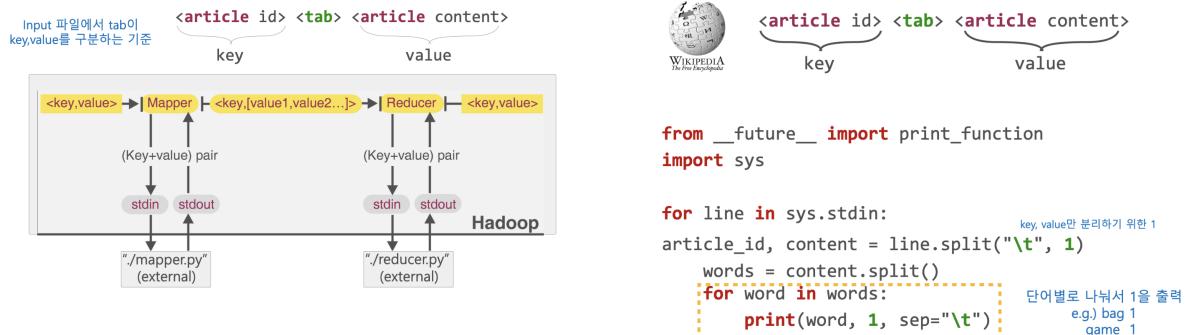
permissions number_of_replicas userid groupid filesize modification_date modification_time filename

```
$ hdfs dfs -text wc_mr_with_reducer/*
--
```

- generic options (job meta configuration parameters..) 다음 command options (mapper or reducer to use)



WordCount in Python



```

yarn jar $HADOOP_STREAMING_JAR \
    -files mapper.py \
    -mapper 'python mapper.py' \
    -numReduceTasks 0 \
    -input /data/wiki/en_articles \
    -output word_count

$ hdfs dfs -text /data/wiki/en_articles/* | head -c 80
12 <tab> Anarchism      Anarchism is often defined as a political
philosophy which ...

$ hdfs dfs -ls -h word_count
Found 3 items
-rw-r--r-- 3 adral adral 0 2017-03-22 11:40 word_count/_SUCCESS
-rw-r--r-- 3 adral adral 47.8 M 2017-03-22 11:40 word_count/part-00000
-rw-r--r-- 3 adral adral 47.9 M 2017-03-22 11:40 word_count/part-00001

```

```

yarn jar $HADOOP_STREAMING_JAR \
    -files mapper.py \
    -mapper 'python mapper.py' \
    -numReduceTasks 0 \
    -input /data/wiki/en_articles \
    -output word_count

$ hdfs dfs -text /data/wiki/en_articles/*
| head -c 80
12 <tab> Anarchism      Anarchism is often defined as a political philosophy
which ...
$ hdfs dfs -text
word_count/part-* | head -5

```

part-00000	part-00001
Basel 1	Anarchism 1
Basel 1	Anarchism 1
(1	is 1
) 1	often 1
or 1	defined 1
...	...

```
yarn jar $HADOOP_STREAMING_JAR \
    -files mapper.py \
    -mapper 'python mapper.py' \
    -numReduceTasks 1 \
    -input /data/wiki/en_articles \
    -output word_count
```

```
$ hdfs dfs -text word_count/part-00000 | head
!
!   1
!
!   1
!
!   1
!
!   1
...
...
```

mapper만 실행해도 numReduceTasks를 1이상으로 선언하면
Shuffle&sort가 수행된다.

key별로 정렬이 된 것을 확인 가능

`mapper.py` 수정 버전

```
<article id> <tab> <article content>
key           value

from __future__ import print_function
import re
import sys

for line in sys.stdin:
    article_id, content = line.split("\t", 1)
    words = re.split("\W+", content) 특수문자를 지우기 위한 정규식 처리
    for word in words:
        if word:
            print(word, 1, sep="\t")
```

```
yarn jar $HADOOP_STREAMING_JAR \
    -files mapper.py \
    -mapper 'python mapper.py' \
    -numReduceTasks 1 \
    -input /data/wiki/en_articles \
    -output word_count
```

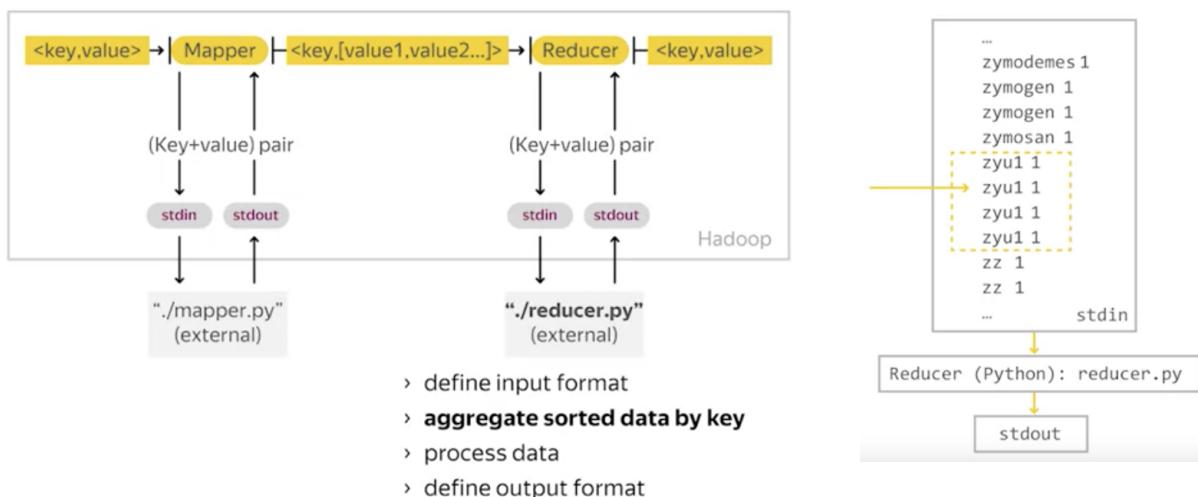
특수문자가 제거됨

```
$ hdfs dfs -text word_count/part-00000 | head -4
0 1
0 1
0 1
0 1
```

```
$ hdfs dfs -tail word_count/part-00000 | tail -4
zyu1 1
zyu1 1
zz 1
zz 1
```

\W: 특수문자 제거

\w: 문자 제거..



```
yarn jar $HADOOP_STREAMING_JAR \
    -files mapper.py,reducer.py \
    -mapper 'python mapper.py' \
    -reducer 'python reducer.py' \
    -numReduceTasks 1 \
    -input /data/wiki/en_articles \
    -output word_count
```

```
$ hdfs dfs -ls -h word_count
Found 2 items
-rw-r--r-- 3 adral adral 0 2017-03-22 13:05 word_count/_SUCCESS
-rw-r--r-- 3 adral adral 3.2 M 2017-03-22 13:05 word_count/part-00000
```

```
$ hdfs dfs -ls -h word_count
Found 2 items
-rw-r--r-- 3 adral adral 0 2017-03-22 13:05 word_count/_SUCCESS
-rw-r--r-- 3 adral adral 3.2 M 2017-03-22 13:05 word_count/part-00000
```

```
$ hdfs dfs -text word_count/part-00000
```

```
0 14905
```

```
00 844
```

```
000 8186
```

```
--
```

```
zymodemes 1
```

```
zymogen 2
```

```
zimosan 1
```

```
zyu1 4
```

```
zz 2
```

```
-- zymosan 1
zyu1 1
zyu1 1
zyu1 1
zyu1 1
zz 1
```

```
yarn jar $HADOOP_STREAMING_JAR \
    -files mapper.py,reducer.py \
    -mapper 'python mapper.py' \
    -reducer 'python reducer.py' \
    -input /data/wiki/en_articles \
    -output word_count
```

```
$ hdfs dfs -ls -h word_count
Found 11 items
-rw-r--r-- 3 adral adral 0 2017-03-22 13:19 word_count/_SUCCESS
-rw-r--r-- 3 adral adral 331.8 K 2017-03-22 13:18 word_count/part-00000
-rw-r--r-- 3 adral adral 332.1 K 2017-03-22 13:18 word_count/part-00001
-rw-r--r-- 3 adral adral 331.7 K 2017-03-22 13:18 word_count/part-00002
-rw-r--r-- 3 adral adral 329.8 K 2017-03-22 13:18 word_count/part-00003
-rw-r--r-- 3 adral adral 326.1 K 2017-03-22 13:18 word_count/part-00004
-rw-r--r-- 3 adral adral 332.2 K 2017-03-22 13:18 word_count/part-00005
-rw-r--r-- 3 adral adral 332.3 K 2017-03-22 13:18 word_count/part-00006
```

- numReduceTasks 지정하지 않으면 자동으로 여러개 생성

```
$ hdfs dfs -tail word_count/part-... | tail -5
```

→ 각각의 reducer에서는 결과가 정렬되지만 globally sort는 되지 않음

⇒ TotalOrderPartitioner 옵션 존재

- › You know **what** MapReduce Streaming **is** and how it works
- › You know **how to write** MapReduce Bash and Python Streaming applications
- › You should be able **to solve** WordCount or similar problems in MapReduce in Python **by yourself**

Distributed Cache

worker들이 사용해야 할 파일을 미리 Node에 업로드해서 공유

```
yarn jar $HADOOP_STREAMING_JAR \
    -files mapper.py,reducer.py,vocabulary.txt \
    -mapper 'python mapper.py' \
    -reducer 'python reducer.py' \
    -input /data/wiki/en_articles \
    -output word_count
```

단어 파일을 worker Node에 업로드

```

from __future__ import print_function
import re
import sys

def read_vocabulary(file_path):
    return set(word.strip() for word in open(file_path))

vocabulary = read_vocabulary("vocabulary.txt")

for line in sys.stdin:
    article_id, content = line.split("\t", 1)
    words = re.split("\W+", content)
    for word in words:
        if word in vocabulary:
            print(word, 1, sep="\t")

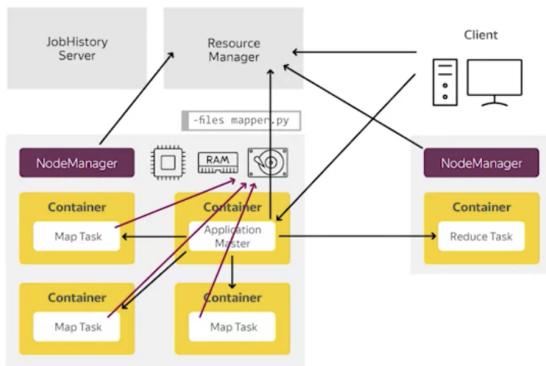
```

```

Mapper (Python): mapper.py
def read_vocabulary(file_path):
    return set(word.strip() for word in open(file_path))

vocabulary = read_vocabulary("vocabulary.txt")

```



- files에 지정된 파일들을 모든 worker node에 업로드함
- task container들은 node의 디스크에 접근하여 데이터를 읽음

Do NOT MODIFY a file content in a distributed cache!

every other container on the same host machine will access the modified version of this file (they are accessed via symlinks). Therefore, if your script execution relies on it then you can get a different output. It breaks functional paradigm and, therefore, you will have non reproducible results.



-files

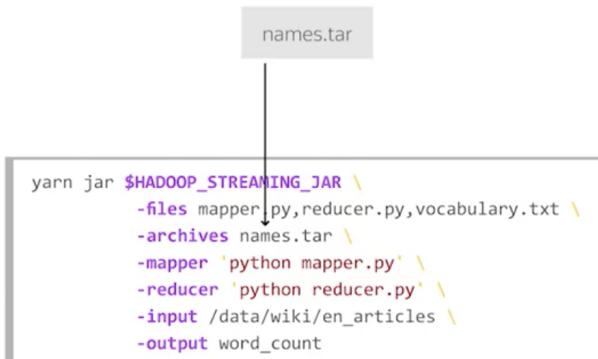


-archives



-libjars

```
$ tar -cf names.tar male.txt female.txt
```

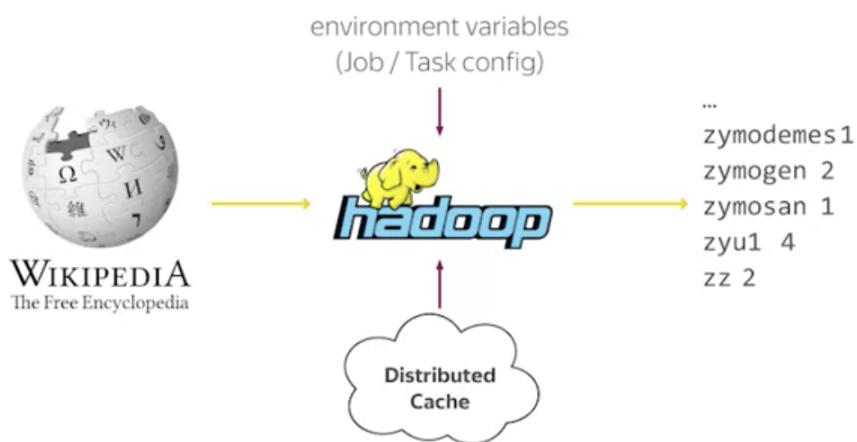


```
$ hdfs dfs -text word_count/* | sort -nrk2,2
James 2284
Thomas 1941
Jack 786
Harry 504
Muhammad 444
Oliver 250
Charlie 250
Jacob 234
Emily 128
Isabella 99
Sophia 92
Noah 80
Sophie 64
Lily 31
Olivia 27
Ethan 27
Ella 25
Amelia 25
Isabelle 18
Chloe 9
```

```
$ hdfs dfs -text word_count/* | sort -nrk2,2
```

Environment, Counters

environment variable also possible



```

Mapper (Python): mapper.py
from __future__ import print_function
import re
import sys

if os.environ["mapred_task_is_map"] == "true":
    print("input_file:{}, start:{}, size:{}".
          format(
              os.environ["mapreduce_map_input_file"],
              os.environ["mapreduce_map_input_start"],
              os.environ["mapreduce_map_input_length"],
          ))

for line in sys.stdin:
    pass

```

default 환경변수 예시
environment variables (Job / Task config)

```

Mapper (Python): mapper.py
os.environ["mapreduce_task_id"]
os.environ["mapreduce_task_partition"]

```

Map Tasks for job_1488734338480_1443

Name	State	Start Time	Finish Time	Elapsed Time
task_1488734338480_1443_m_000001	SUCCEEDED	Sat Mar 25 19:57:01 +0000 2017	Sat Mar 25 19:57:05 +0000 2017	4sec
task_1488734338480_1443_r_000008	SUCCEEDED	Sat Mar 25 19:57:01 +0000 2017	Sat Mar 25 19:57:05 +0000 2017	4sec

```

from __future__ import print_function
import re
import sys

CHARS_IN_LINE = 9

if os.environ["mapred_task_is_map"] == "true":
    split_input_start = int(
        os.environ["mapreduce_map_input_start"])
    //CHARS_IN_LINE

for split_line_index, line in enumerate(sys.stdin):
    line_number = split_line_index + split_input_start
    if (line_number < 10):
        print(line_number, line, sep='\t')

```



WIKIPEDIA
The Free Encyclopedia

`<article id> <tab> <article content>`

Mapper (Python): wc_mapper.py

```

from __future__ import print_function
import os
import re
import sys

유저가 생성한 환경변수
pattern = re.compile(os.environ["word_pattern"])

for line in sys.stdin:
    article_id, content = line.split("\t", 1)
    words = re.findall(pattern, content)
    for word in words:
        print(word, 1, sep="\t")

```

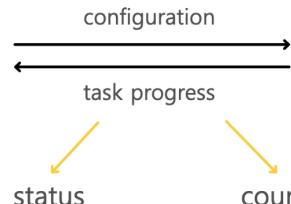
유저가 생성한 환경변수 -D 를 활용

```

yarn jar $HADOOP_STREAMING_JAR -D word_pattern="\w+\d+" \
    -files mapper.py \
    -mapper 'python mapper.py' \
    -reducer 'python reducer.py' \
    -input /data/wiki/en_articles \
    -output word_count

$ hdfs dfs -text word_count/*
...
test2 4
times11 1
times48 3
tinctorial 1
titan2
...

```



반대로 Script 코드에서 Hadoop에게 status와 counters 값을 전달할 수 있다.

Status example

Show 20+ entries		
Attempt	State	Status
attempt_1488734338480_1448_m_000000_0	SUCCEEDED	processed 6374 words
attempt_1488734338480_1448_m_000001_0	SUCCEEDED	processed 1778 words
Showing 1 to 2 of 2 entries		

Mapper (Python): reporter_mapper.py

```
from __future__ import print_function
import re
import sys

for line in sys.stdin:
    article_id, content = line.split("\t", 1)
    words = re.findall("\w+", content)
    for index, word in enumerate(words):
        print(word, 1, sep="\t") 현재 worker의 상태를 표준출력으로 전달한다.
        print("reporter:status:processed {} words"
              .format(index + 1), file=sys.stderr)
```

Mapper (Python): reporter_mapper.py

```
from __future__ import print_function
import re
import sys

for line in sys.stdin:
    article_id, content = line.split("\t", 1)
    words = re.findall("\w+", content)
    for index, word in enumerate(words):
        print(word, 1, sep="\t")
        print("reporter:status:processed {} words"
              .format(index + 1), file=sys.stderr)
    print("reporter:counter:Personal Counters,word found,1", file=sys.stderr)
```

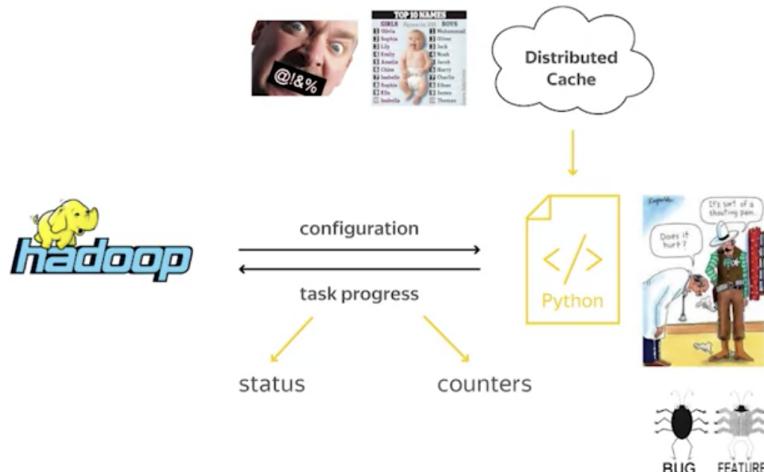
Counter example

reporter:counter:<group>,<counter>,<amount>	↑	↑	↑
Map-Reduce Framework	Map output materialized bytes	954408	0
	Merged Map outputs	12396473	20
	Physical memory (bytes) snapshot	2732658464	6125685832
	Reduce input records	308456	308456
	Reduce input records	0	12396473
	Reduce shuffle bytes	0	308456
	Shuffled Maps	0	954408
	Total committed heap usage (bytes)	12396473	237293464
	Total committed heap usage (bytes)	9422281736	624321504
	Virtual memory (bytes) snapshot	7705353096	8320534580
Personal Counters	word found	12396473	0
	Map	0	12396473
	Reduce	0	0
	Total	0	12396473

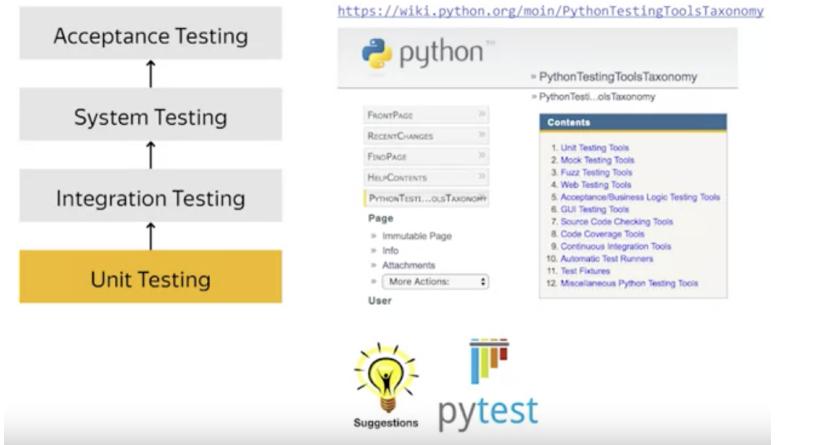
> You know how to:

- > provide environment variables (global configuration) to your streaming scripts
- > access job configuration options (e.g. map input file)
- > report progress back to Hadoop MapReduce Framework

Testing



- Unit Testing - debug by python libraries



```
def get_words(input_line):
    return input_line.split()

def test_get_words_parse_simple_string():
    assert get_words("a b cd efg ") == ["a", "b", "cd", "efg"]

def test_get_words_parse_empty_string():
    assert get_words("") == []

def test_get_words_raise_exception_if_no_input():
    with pytest.raises(AttributeError):
        get_words(None)
```

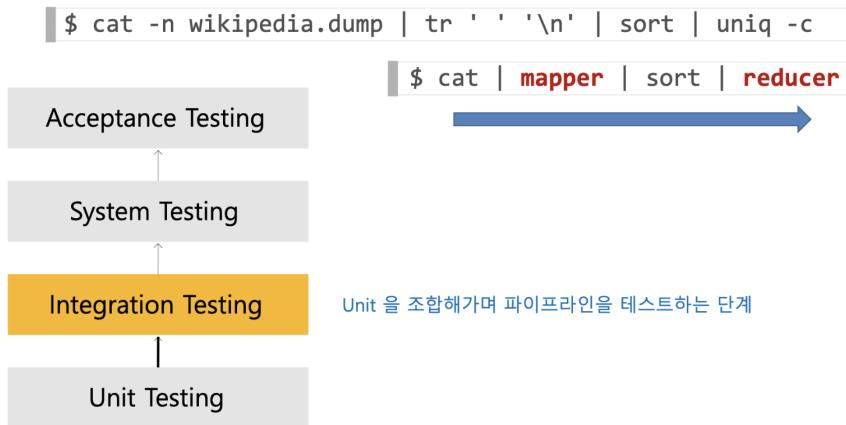
원치 않는 단어, 빈 값, input 없는 경우

```
(venv) > courses pytest -v test_function.py
platform darwin -- Python 2.7.12, pytest-3.6.7, py-1.4.33, pluggy-0.4.0 -- /Users/eddal/workspace/personal/mipt/courses/venv/bin/python2.7
cachedir: .cache
rootdir: /Users/eddal/workspace/personal/mipt/courses, inifile:
collected 3 items

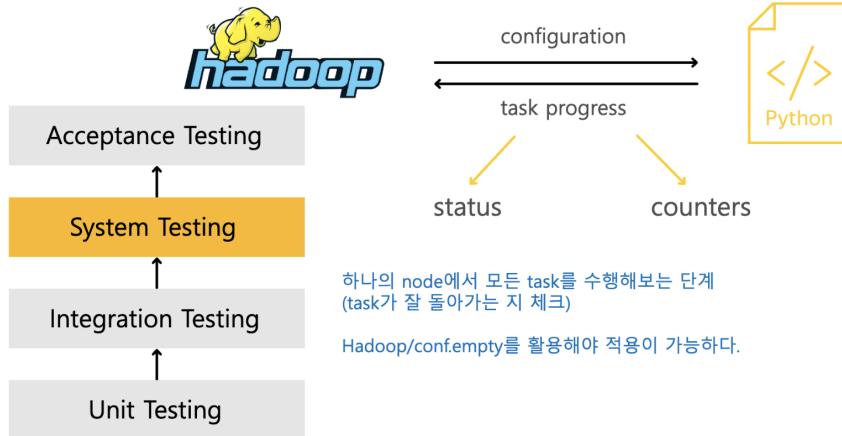
test_function.py::test_get_words_parse_simple_string PASSED
test_function.py::test_get_words_parse_empty_string PASSED
test_function.py::test_get_words_raise_exception_if_no_input PASSED

* passed in 0.01 seconds
```

- **Integration Testing - Unit 조합해가며 파이프라인을 테스트 (emulate mapreduce locally with cat|map|sort|reduce)**



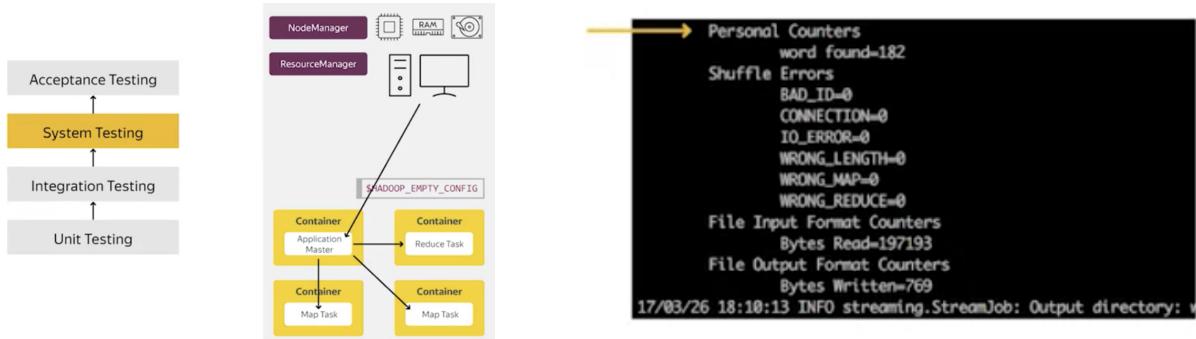
- **System Testing - 하나의 node에서 모든 task 수행해봄 (run mapreduce in a standalone mode)**



```
hdfs --config $HADOOP_EMPTY_CONFIG dfs -rm -r word_count
yarn --config $HADOOP_EMPTY_CONFIG jar $HADOOP_STREAMING_JAR
```

```
$ locate "hadoop/conf.empty"
```

```
$ find "hadoop/conf.empty"
$ locate "hadoop/conf.empty"
```



- **Acceptance Testing - why you need to execute mapreduce against sample datasets**



MapReduce Streaming (Quiz).