Working with Hadoop

- Hadoop natively written in Java and hence supports Java for deep customisation work
- However, it enables interfaces to Python, R and other languages using a concept called Hadoop Streaming
- We will work with Python on Hadoop
- Possible to work directly using Python on Hadoop as Hadoop natively supports it but it is recommended to use certain python framework for ease of work without big sacrifice in terms of features
- We will use MRJob framework - https://pythonhosted.org/mrjob/

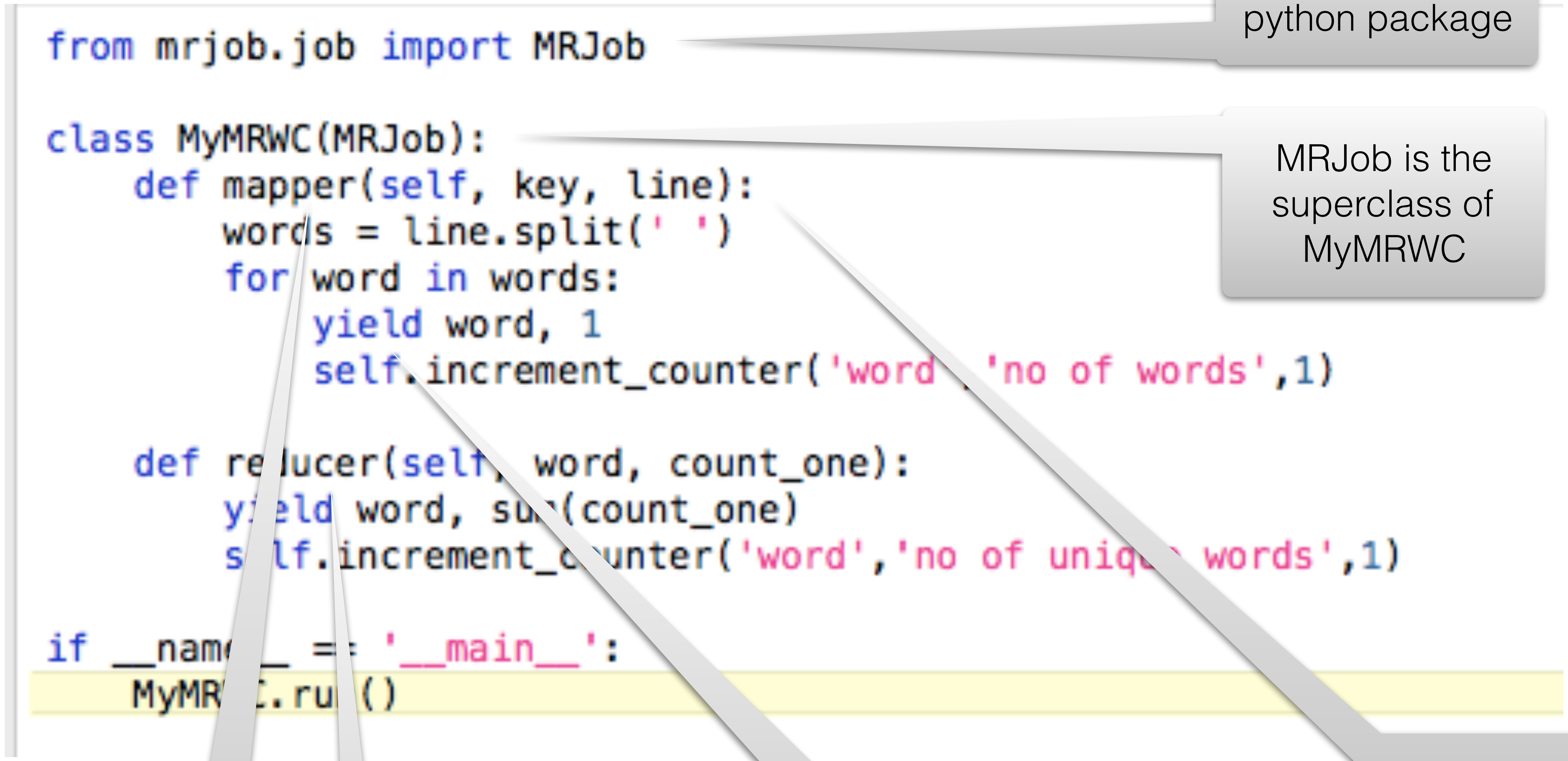
Hadoop Modes

- Local Mode - single process space (we will use this as this mode is good for learning, experimenting, coding) - any code which runs well in this mode is *almost* guaranteed to run in the other modes
- Pseudo Mode - simulation of multiple process space through the use of host:port combination although in reality only one node (single node cluster)
- Cluster Mode - proper multi-node cluster with single master server and 1 or more slave nodes

Map Reduce

- http://hadoop.apache.org/docs/current/hadoop-mapreduce-client/hadoop-mapreduce-client-core/MapReduceTutorial.html

- Mapper program
- Reducer program
- Counters



```
from mrjob.job import MRJob

class MyMRWC(MRJob):
    def mapper(self, key, line):
        words = line.split(' ')
        for word in words:
            yield word, 1
            self.increment_counter('word','no of words',1)

    def reducer(self, word, count_one):
        yield word, sum(count_one)
        self.increment_counter('word','no of unique words',1)

if __name__ == '__main__':
    MyMRWC.run()
```

import the appropriate python package

MRJob is the superclass of MyMRWC

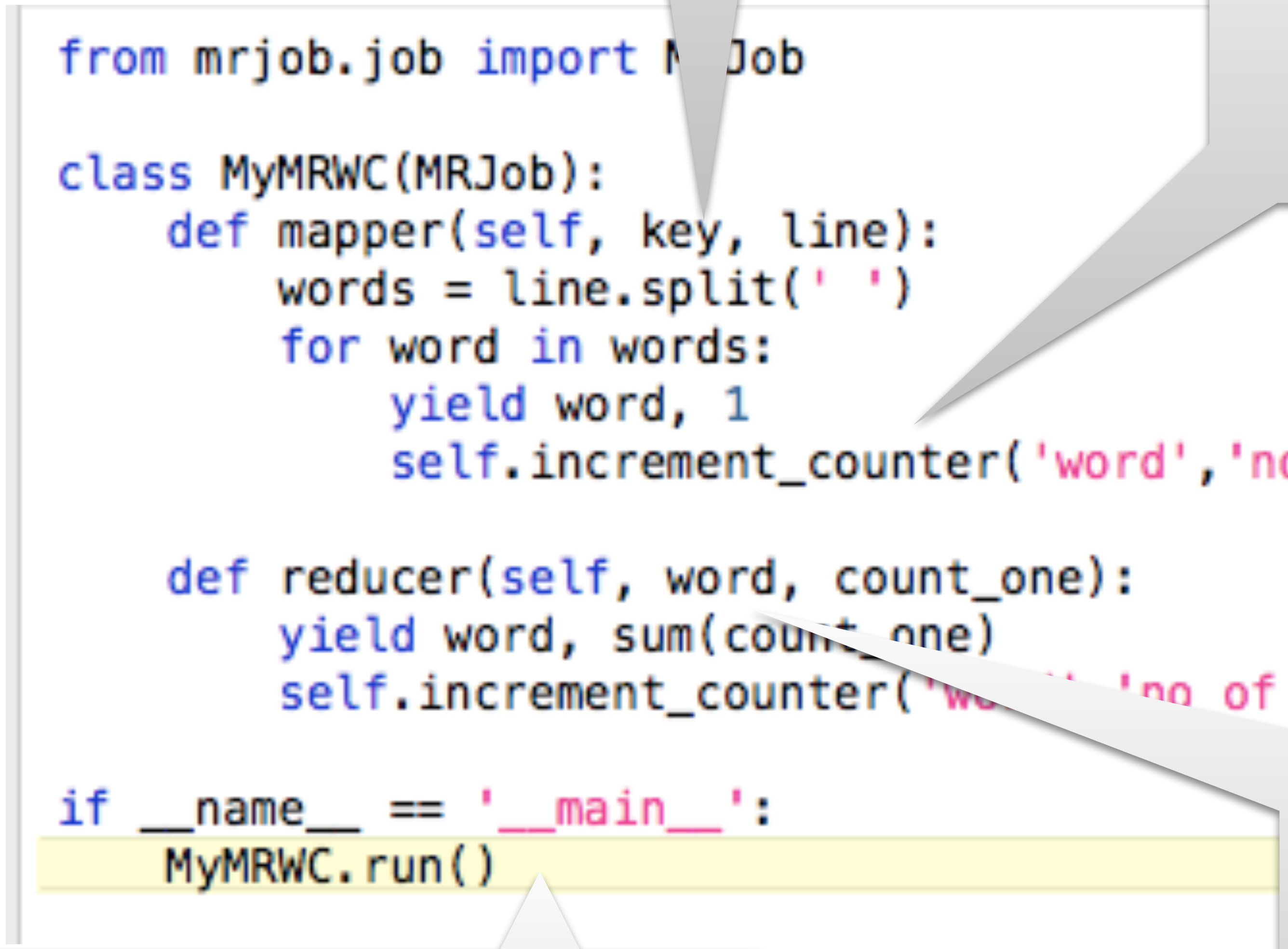
First param is self in python

yield in python means return of a value; there can be multiple yields in the same function

standard name of the mapper function "mapper"

standard name of the reducer function "reducer" "key" in this case is NoneType (null) and "line" is the value associated with the line number (names are not standard but positions are) - input to mapper function is hence a key-value pair

this is usage of counter to keep track of things across the entire cluster - we will discuss it later

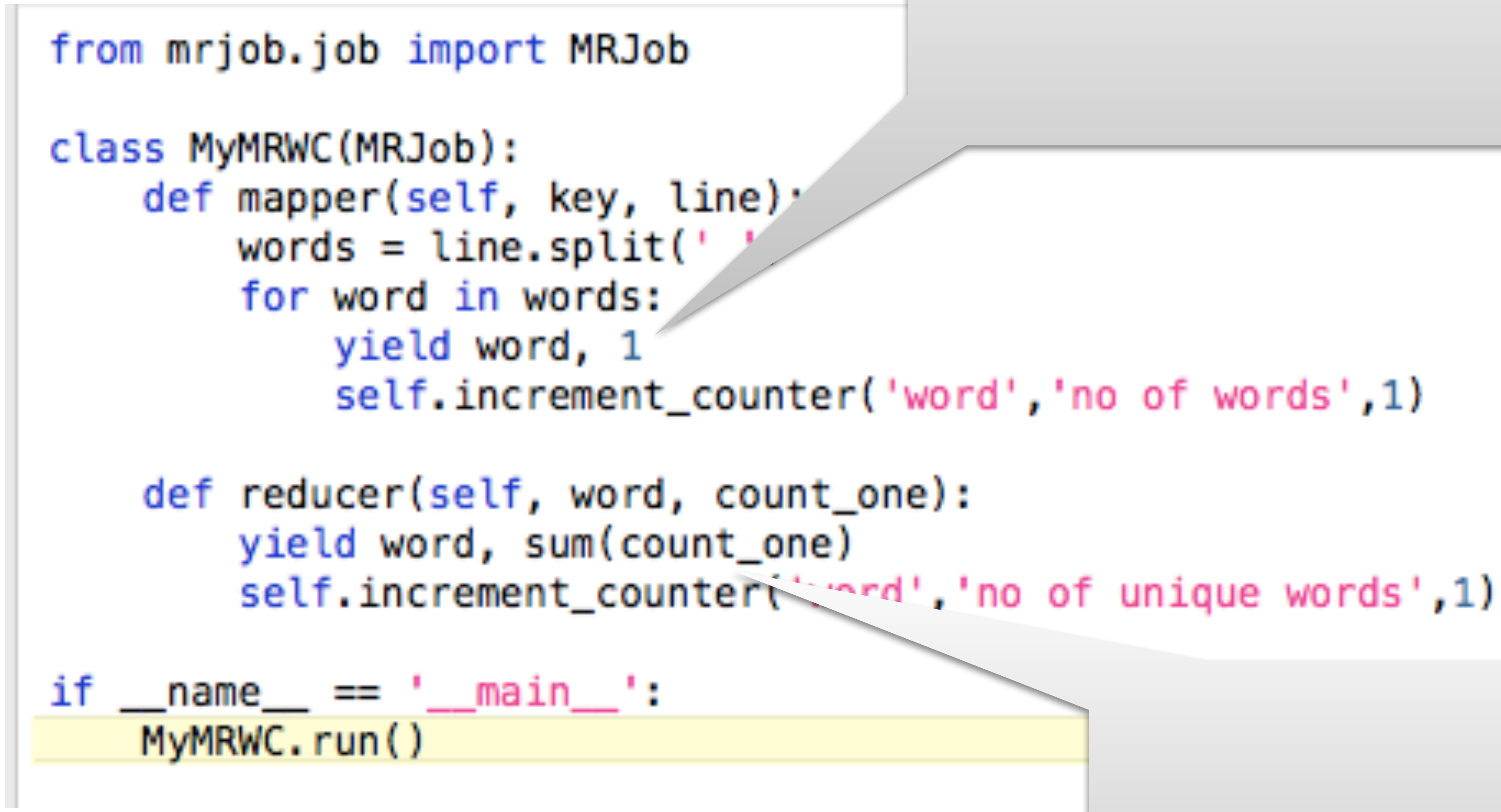


```
from mrjob.job import MRJob

class MyMRWC(MRJob):
    def mapper(self, key, line):
        words = line.split(' ')
        for word in words:
            yield word, 1
            self.increment_counter('word','no of words',1)

    def reducer(self, word, count_one):
        yield word, sum(count_one)
        self.increment_counter('word','no of unique words',1)

if __name__ == '__main__':
    MyMRWC.run()
```

"word" in this case is the key and "count_one" is the value associated with the key (names are not standard but positions are) - input to reducer function is hence a key-value pair

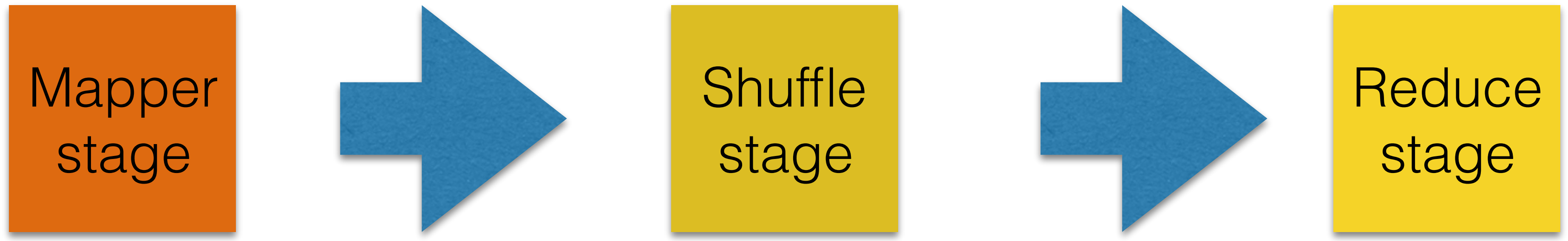
No need to worry about this block of code; ensure it is there in all your python programs; it ensures that the program can be run only as a main program; just ensure that the classname is correctly used before run( )

the mapper function always returns a pair (key-value pair)



```
from mrjob.job import MRJob

class MyMRWC(MRJob):
    def mapper(self, key, line):
        words = line.split(' ')
        for word in words:
            yield word, 1
            self.increment_counter('word','no of words',1)

    def reducer(self, word, count_one):
        yield word, sum(count_one)
        self.increment_counter('word','no of unique words',1)

if __name__ == '__main__':
    MyMRWC.run()
```

the reducer function always returns a pair (key-value pair)



For Each operation

Group By operation

Aggregate operation - sum, average, etc.

Map

- Mapper function runs once per line of the input file
- The function transforms the input K1-V1 pair to another K2-V2 pair or more such pairs
- V1 = (1212, 23462) => K2 = Ravi, V2 = 100 and may be also K3 = Kolkata, V3 = 1

Shuffle

- This is the most important step which brings together (group by) all the values for the same key at one place
- It receives output from map and forwards the output to usually to a reduce stage

Reduce

- Reducer function receives a key-list pair
- It runs once for each key
- The reducer function should be created so that it can handle list (collection of items) type of data
-