Tamkang University
淡江大學

Tamkang University

Social Computing and Big Data Analytics
社群運算與大數據分析

Big Data Processing Platforms with SMACK: Spark, Mesos, Akka, Cassandra and Kafka
(大數據處理平台SMACK)

1042SCBDA04
MIS MBA (M2226) (8628)
Wed, 8,9, (15:10-17:00) (B309)

Min-Yuh Day
戴敏育
Assistant Professor
專任助理教授
Dept. of Information Management, Tamkang University
淡江大學 資訊管理學系
http://mail. tku.edu.tw/myday/

2016-03-09

課程大綱 (Syllabus)

週次 (Week)	日期 (Date)	內容 (Subject/Topics)
1	2016/02/17	Course Orientation for Social Computing and Big Data Analytics (社群運算與大數據分析課程介紹)
2	2016/02/24	Data Science and Big Data Analytics: Discovering, Analyzing, Visualizing and Presenting Data (資料科學與大數據分析：探索、分析、視覺化與呈現資料)
3	2016/03/02	Fundamental Big Data: MapReduce Paradigm, Hadoop and Spark Ecosystem (大數據基礎：MapReduce典範、Hadoop與Spark生態系統)

課程大綱 (Syllabus)

週次 (Week)	日期 (Date)	內容 (Subject/Topics)
4	2016/03/09	Big Data Processing Platforms with SMACK: Spark, Mesos, Akka, Cassandra and Kafka (大數據處理平台SMACK：Spark, Mesos, Akka, Cassandra, Kafka)
5	2016/03/16	Big Data Analytics with Numpy in Python (Python Numpy 大數據分析)
6	2016/03/23	Finance Big Data Analytics with Pandas in Python (Python Pandas 財務大數據分析)
7	2016/03/30	Text Mining Techniques and Natural Language Processing (文字探勘分析技術與自然語言處理)
8	2016/04/06	Off-campus study (教學行政觀摩日)

課程大綱 (Syllabus)

週次 (Week)	日期 (Date)	內容 (Subject/Topics)
9	2016/04/13	Social Media Marketing Analytics (社群媒體行銷分析)
10	2016/04/20	期中報告 (Midterm Project Report)
11	2016/04/27	Deep Learning with Theano and Keras in Python (Python Theano 和 Keras 深度學習)
12	2016/05/04	Deep Learning with Google TensorFlow (Google TensorFlow 深度學習)
13	2016/05/11	Sentiment Analysis on Social Media with Deep Learning (深度學習社群媒體情感分析)

課程大綱 (Syllabus)

週次 (Week)	日期 (Date)	內容 (Subject/Topics)
14	2016/05/18	Social Network Analysis (社會網絡分析)
15	2016/05/25	Measurements of Social Network (社會網絡量測)
16	2016/06/01	Tools of Social Network Analysis (社會網絡分析工具)
17	2016/06/08	Final Project Presentation I (期末報告 I)
18	2016/06/15	Final Project Presentation II (期末報告 II)

2016/03/09

Big Data Processing Platforms with SMACK:

Spark, Mesos, Akka, Cassandra and Kafka

(大數據處理平台SMACK：Spark, Mesos, Akka, Cassandra, Kafka)

SMACK Architectures

Building data processing platforms with Spark, Mesos, Akka, Cassandra and Kafka

Source: Anton Kirillov (2015), Data processing platforms architectures with Spark, Mesos, Akka, Cassandra and Kafka, Big Data AW Meetup

SMACK Stack

- **Spark**
  - fast and general engine for distributed, large-scale data processing

- **Mesos**
  - cluster resource management system that provides efficient resource isolation and sharing across distributed applications

- **Akka**
  - a toolkit and runtime for building highly concurrent, distributed, and resilient message-driven applications on the JVM

- **Cassandra**
  - distributed, highly available database designed to handle large amounts of data across multiple datacenters

- **Kafka**
  - a high-throughput, low-latency distributed messaging system designed for handling real-time data feeds

Source: Anton Kirillov (2015), Data processing platforms architectures with Spark, Mesos, Akka, Cassandra and Kafka, Big Data AW Meetup

Spark Ecosystem

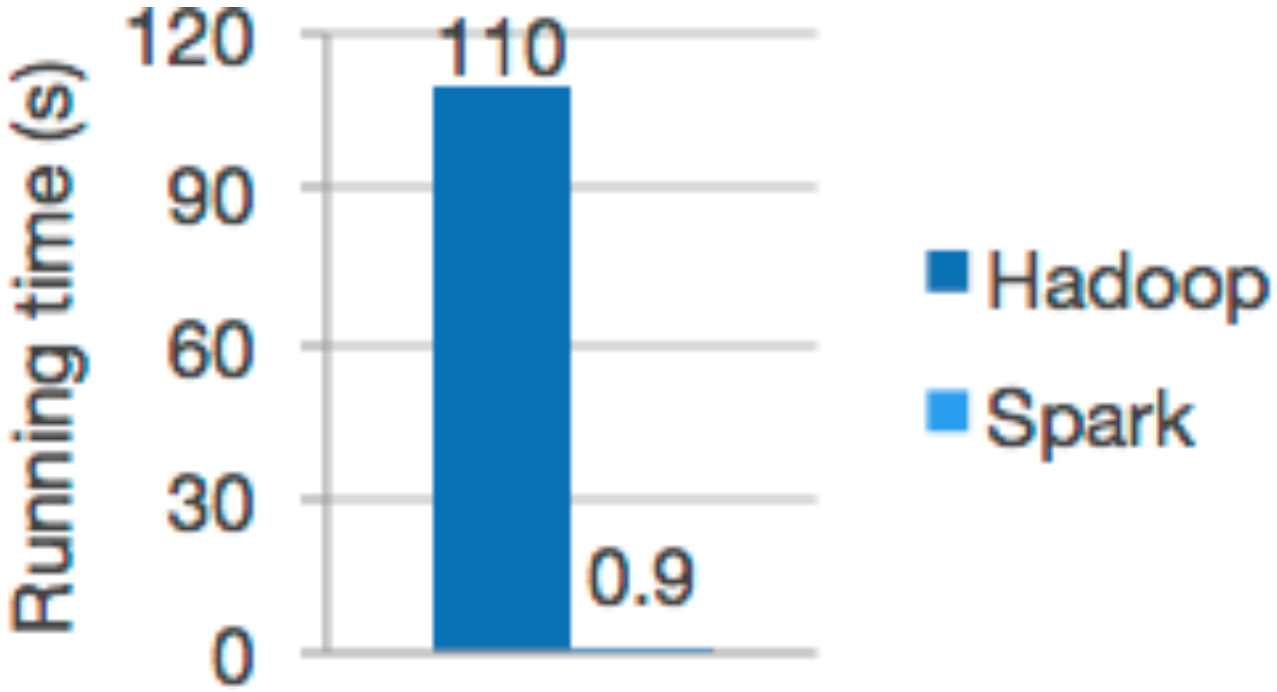
Lightning-fast cluster computing

Apache Spark

is a fast and general engine for large-scale data processing.

Source: http://spark.apache.org/

Logistic regression in Hadoop and Spark



Run programs up to 100x faster than Hadoop MapReduce in memory, or 10x faster on disk.

Source: http://spark.apache.org/

Ease of Use

- Write applications quickly in Java, Scala, Python, R.

Source: http://spark.apache.org/

Word count in Spark's Python API

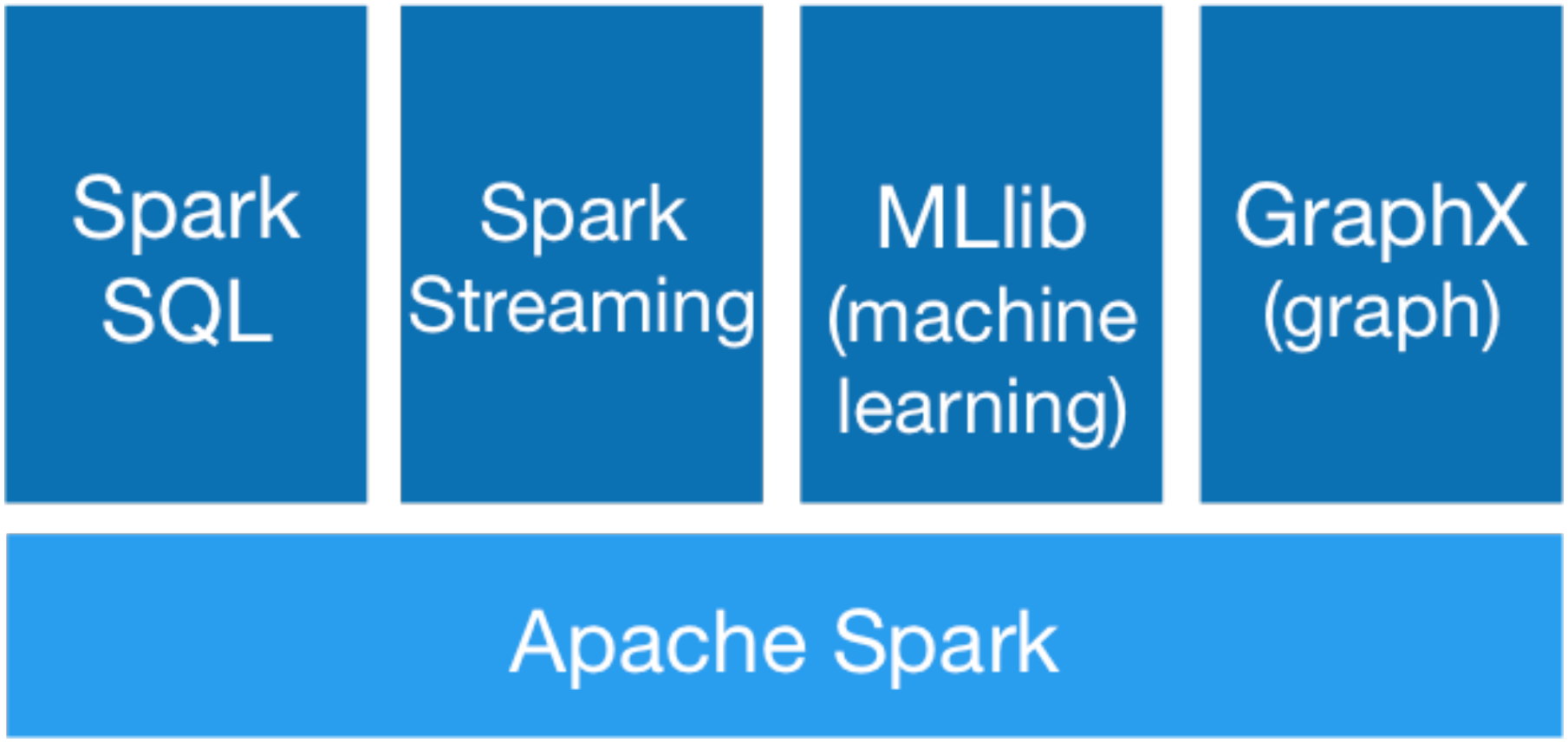
```
text_file = spark.textFile("hdfs://...")

text_file.flatMap(lambda line: line.split())
    .map(lambda word: (word, 1))
    .reduceByKey(lambda a, b: a+b)
```

Source: http://spark.apache.org/

Spark and Hadoop

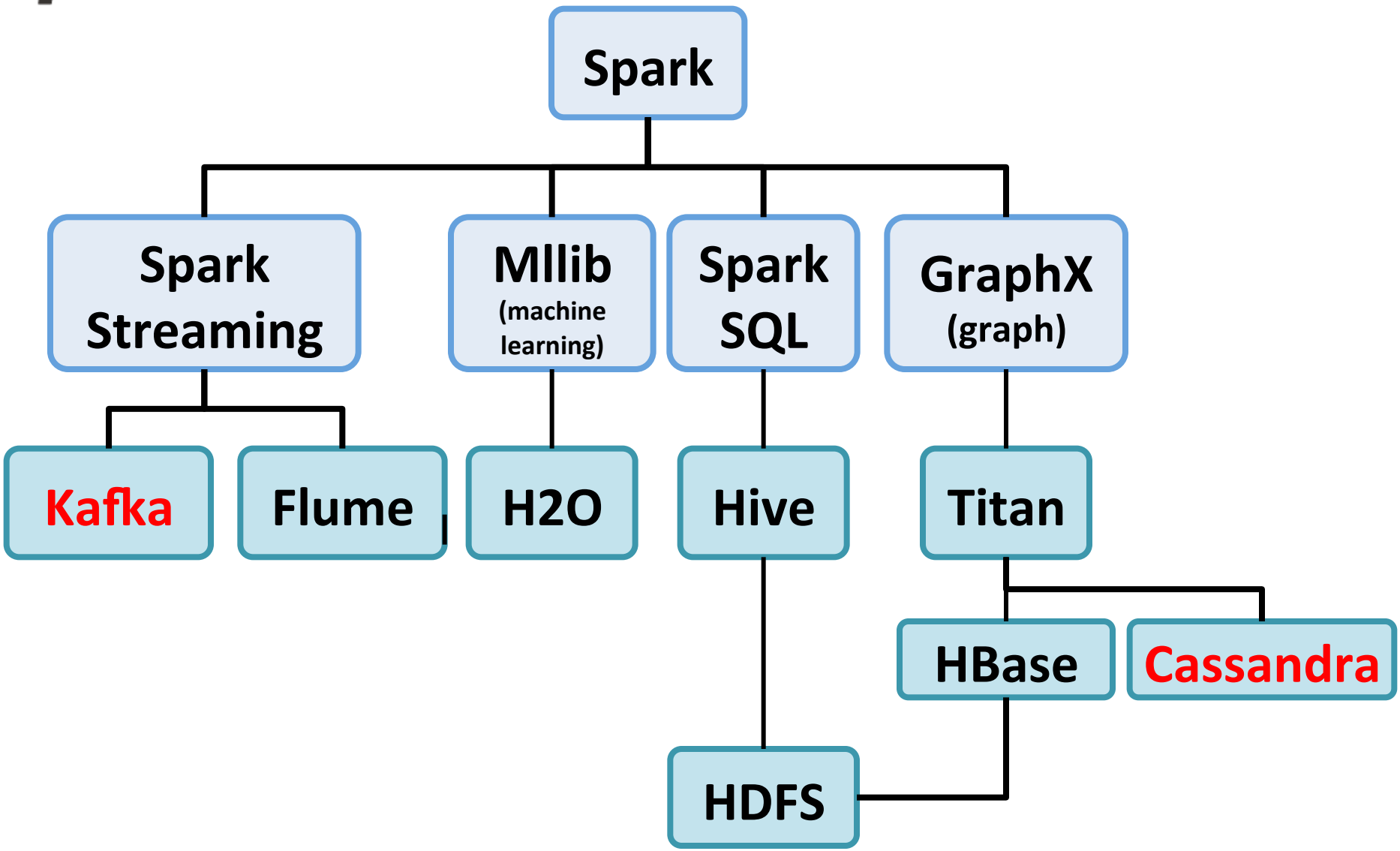
Source: http://spark.apache.org/

Spark Ecosystem



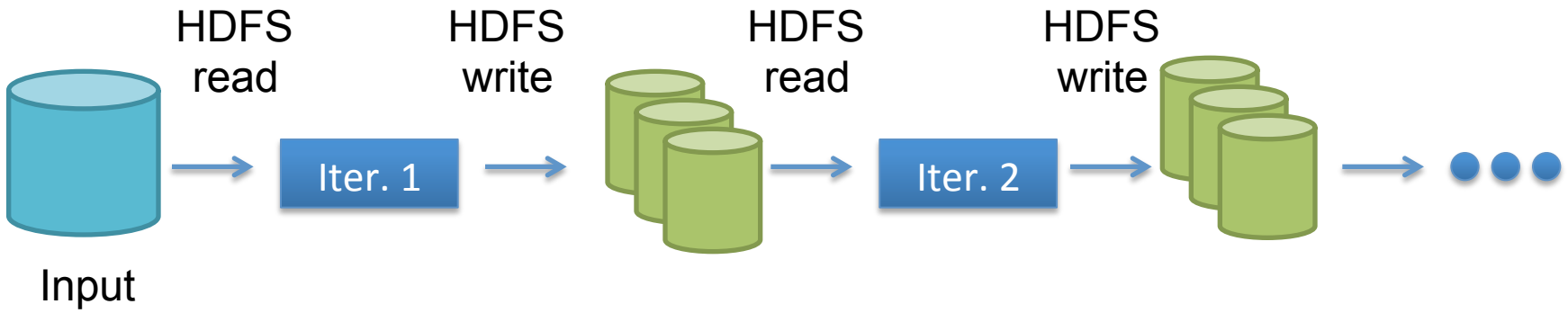
Source: http://spark.apache.org/

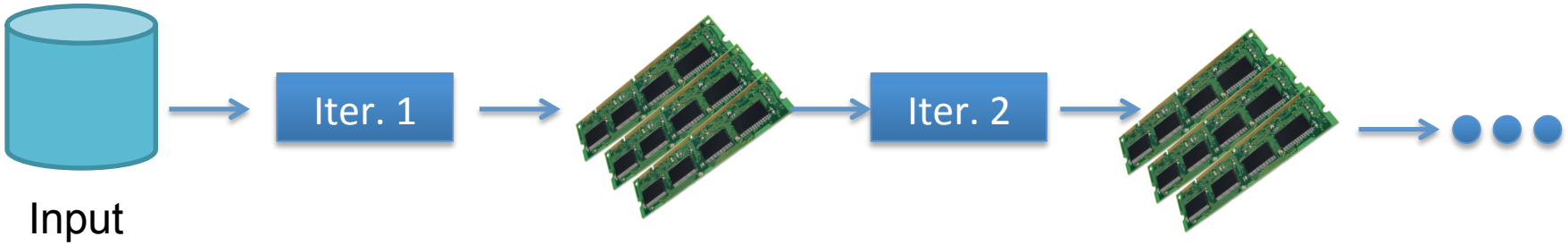
Spark Ecosystem



Source: Mike Frampton (2015), Mastering Apache Spark, Packt Publishing

Hadoop vs. Spark





Source: Shiva Achari (2015), Hadoop Essentials - Tackling the Challenges of Big Data with Hadoop, Packt Publishing

SMACK Stack

- **Spark**
  - fast and general engine for distributed, large-scale data processing

- **Mesos**
  - cluster resource management system that provides efficient resource isolation and sharing across distributed applications

- **Akka**
  - a toolkit and runtime for building highly concurrent, distributed, and resilient message-driven applications on the JVM

- **Cassandra**
  - distributed, highly available database designed to handle large amounts of data across multiple datacenters

- **Kafka**
  - a high-throughput, low-latency distributed messaging system designed for handling real-time data feeds

Source: Anton Kirillov (2015), Data processing platforms architectures with Spark, Mesos, Akka, Cassandra and Kafka, Big Data AW Meetup

Spark

Spark *Lightning-fast cluster computing*

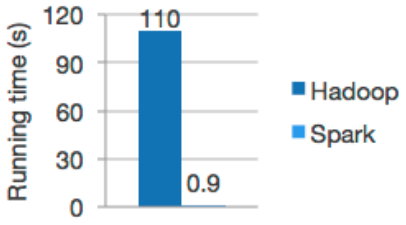
Download | Libraries | Documentation | Examples | Community | FAQ | Apache Software Foundation

Apache Spark™ is a fast and general engine for large-scale data processing.

Speed

Run programs up to 100x faster than Hadoop MapReduce in memory, or 10x faster on disk.

Spark has an advanced DAG execution engine that supports cyclic data flow and in-memory computing.



Logistic regression in Hadoop and Spark

Latest News

Submission is open for Spark Summit San Francisco (Feb 11, 2016)

Spark Summit East (Feb 16, 2016, New York) agenda posted (Jan 14, 2016)

Spark 1.6.0 released (Jan 04, 2016)

CFP for Spark Summit East 2016 is closing soon! (Nov 19, 2015)

Archive

Download Spark

Ease of Use

Write applications quickly in Java, Scala, Python, R.

Spark offers over 80 high-level operators that make it easy to build parallel apps. And you can use it *interactively* from the Scala, Python and R shells.

```
text_file = spark.textFile("hdfs://...")

text_file.flatMap(lambda line: line.split())
    .map(lambda word: (word, 1))
    .reduceByKey(lambda a, b: a+b)
```

Word count in Spark's Python API

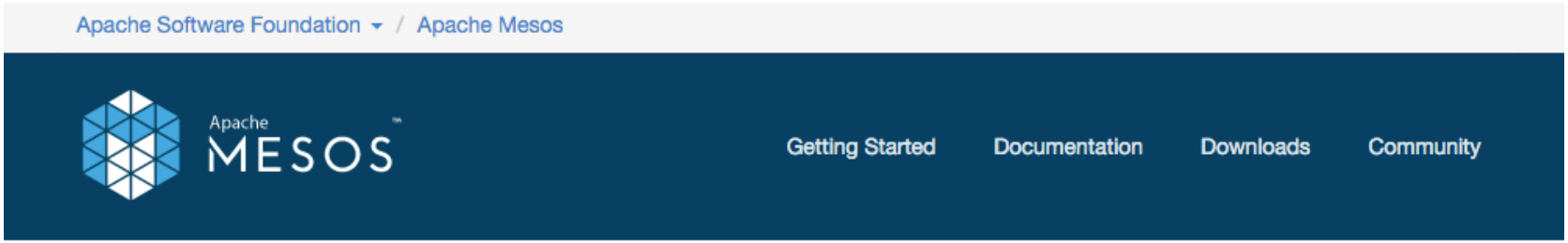
Built-in Libraries:

SQL and DataFrames
Spark Streaming
MLlib (machine learning)
GraphX (graph)

Third-Party Packages

http://spark.apache.org/

Mesos



Program against your datacenter like it's a single pool of resources

Apache Mesos abstracts CPU, memory, storage, and other compute resources away from machines (physical or virtual), enabling fault-tolerant and elastic distributed systems to easily be built and run effectively.

Download Mesos 0.27.2 or learn how to get started

What is Mesos?

A distributed systems kernel

Mesos is built using the same principles as the Linux kernel, only at a different level of abstraction. The Mesos kernel runs on every machine and provides applications (e.g., Hadoop, Spark, Kafka, Elastic Search) with API's for resource management and scheduling across entire datacenter and cloud environments.

Project Features

- Scalability to 10,000s of nodes

News

- *March 7, 2016* - Mesos 0.27.2 is released! See the CHANGELOG and blog post for details.
- *February 22, 2016* - Mesos 0.27.1 is released! See the CHANGELOG and blog post for details.
- *February 12, 2016* - MesosCon 2016 CFP is now open! See the blog post for details.
- *January 31, 2016* - Mesos 0.27.0 is released! See the CHANGELOG and blog post for details.
- *December 16, 2015* - Mesos 0.26.0 is released!

http://mesos.apache.org/

Akka

http://akka.io/

Cassandra

Home Download Getting Started Planet Cassandra Contribute

Welcome Video Slides

Welcome to Apache Cassandra ™

The Apache Cassandra database is the right choice when you need scalability and high availability without compromising performance. Linear scalability and proven fault-tolerance on commodity hardware or cloud infrastructure make it the perfect platform for mission-critical data. Cassandra's support for replicating across multiple datacenters is best-in-class, providing lower latency for your users and the peace of mind of knowing that you can survive regional outages.

Cassandra's data model offers the convenience of column indexes with the performance of log-structured updates, strong support for denormalization and materialized views, and powerful built-in caching.

Download

Tick-Tock release **3.4** (Changes)
3.0.x release **3.0.4** (Changes)
2.2.x release **2.2.5** (Changes)

Download options

Overview

Proven

Cassandra is in use at Constant Contact, CERN, Comcast, eBay, GitHub, GoDaddy, Hulu, Instagram, Intuit, Netflix, Reddit, The

Performant

Cassandra consistently outperforms popular NoSQL alternatives in benchmarks and real applications, primarily because of

You're in Control

Choose between synchronous or asynchronous replication for each update. Highly available asynchronous operations are

http://cassandra.apache.org/

Kafka

Apache Kafka

A high-throughput distributed messaging system.

- download
- introduction
- uses
- documentation
- quickstart
- performance
- clients
- ecosystem
- faq
- project
  - twitter
  - wiki
  - bugs
  - mailing lists
  - committers
  - powered by
  - papers & talks
- developers
  - code
  - projects
  - contributing
  - coding guide

Apache Kafka is publish-subscribe messaging rethought as a distributed commit log.

Fast

A single Kafka broker can handle hundreds of megabytes of reads and writes per second from thousands of clients.

Scalable

Kafka is designed to allow a single cluster to serve as the central data backbone for a large organization. It can be elastically and transparently expanded without downtime. Data streams are partitioned and spread over a cluster of machines to allow data streams larger than the capability of any single machine and to allow clusters of co-ordinated consumers

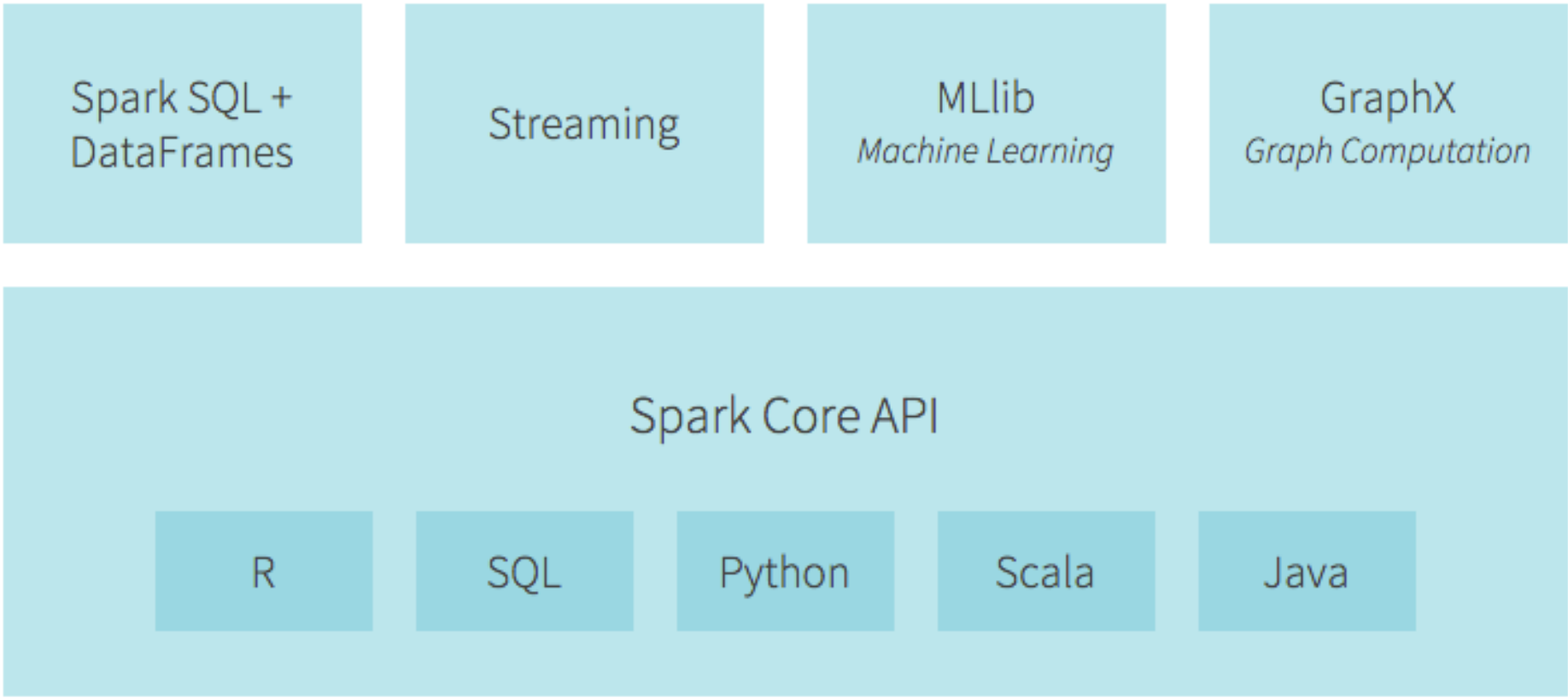
Durable

Messages are persisted on disk and replicated within the cluster to prevent data loss. Each broker can handle terabytes of messages without performance impact.

Distributed by Design

Kafka has a modern cluster-centric design that offers strong durability and fault-tolerance guarantees.

Spark Ecosystem



Source: https://databricks.com/spark/about

databricks

databricks PRODUCT SPARK SOLUTIONS CUSTOMERS COMPANY BLOG RESOURCES Partners Training Sign Up

Apache Spark™ is a powerful open source processing engine built around speed, ease of use, and sophisticated analytics. It was originally developed at UC Berkeley in 2009.

The largest open source project in data processing.

Since its release, Spark has seen rapid adoption by enterprises across a wide range of industries. Internet powerhouses such as Yahoo, Baidu, and Tencent, have eagerly deployed Spark at massive scale, collectively processing multiple petabytes of data on clusters of over 8,000 nodes. It has quickly become the largest open source community in big data, with over 750 contributors from 200+ organizations.

The creators of Spark founded Databricks in 2013.

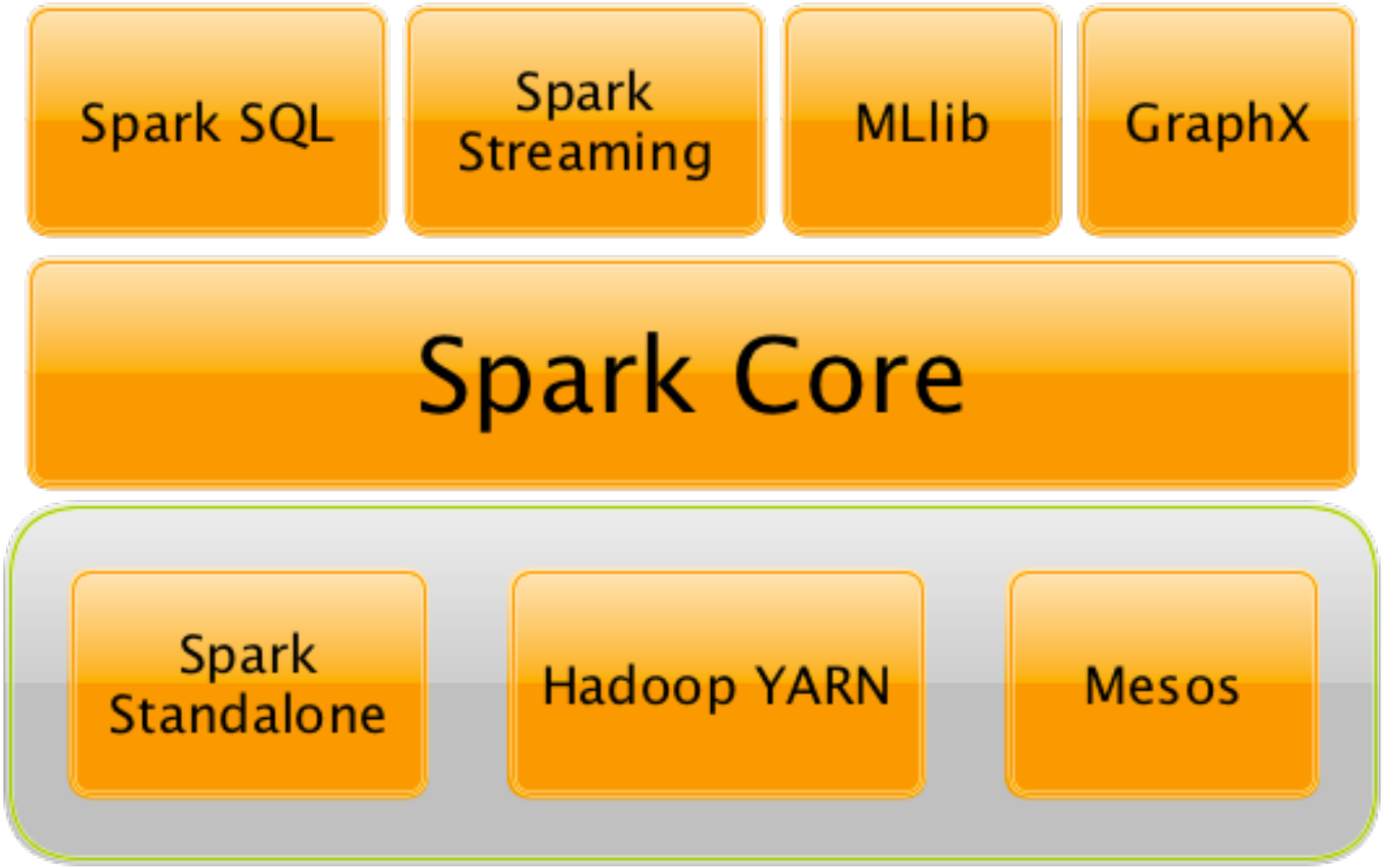
Databricks continues to grow the Spark project by contributing broadly, with both roadmap development and community evangelism.

"At Databricks, we're working hard to make Spark easier to use and run than ever, through our efforts on both the Spark codebase and support materials around it. All of our work on Spark is open source and goes directly to Apache."

— Matei Zaharia, VP, Apache Spark, Founder & CTO, Databricks

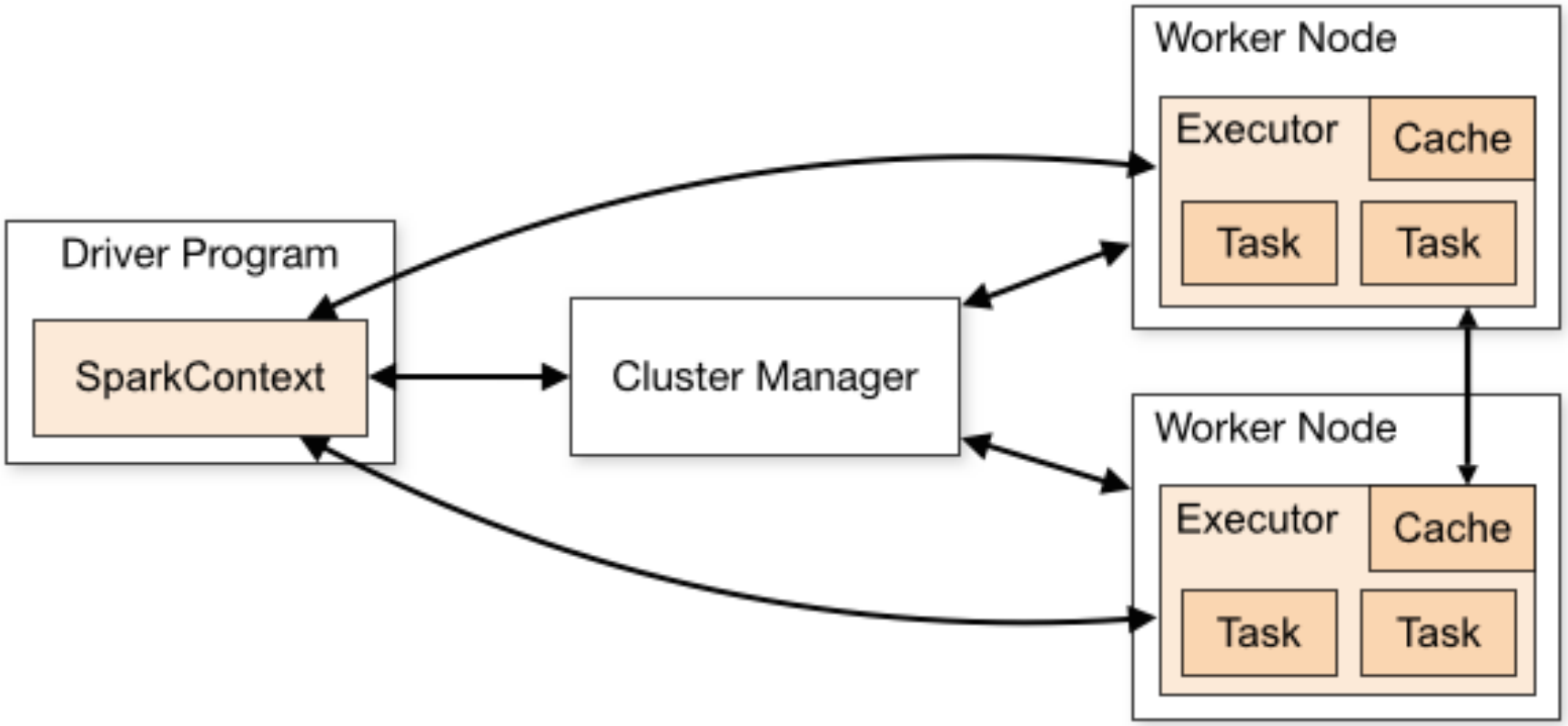
https://databricks.com/spark/about

The Spark Platform



Source: https://jaceklaskowski.gitbooks.io/mastering-apache-spark/content/spark-overview.html

Spark Cluster Overview



Source: http://spark.apache.org/docs/latest/cluster-overview.html

3 Types of Spark Cluster Manager

- **Standalone**
  - a simple cluster manager included with Spark that makes it easy to set up a cluster.
- **Apache Mesos**
  - a general cluster manager that can also run Hadoop MapReduce and service applications.
- **Hadoop YARN**
  - the resource manager in Hadoop 2.

Spark's EC2 launch scripts make it easy to launch a standalone cluster on Amazon EC2.

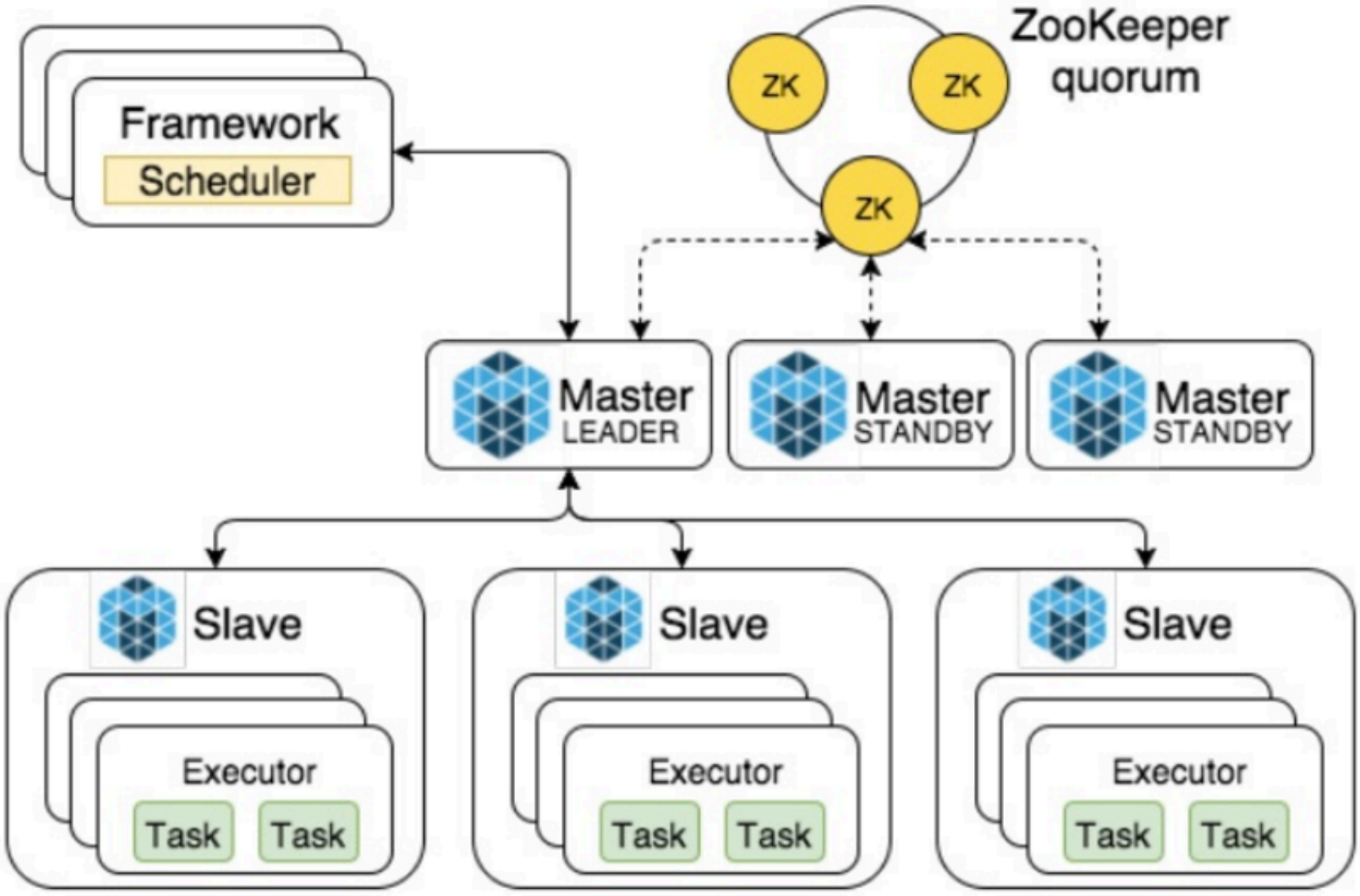
Source: http://spark.apache.org/docs/latest/cluster-overview.html

Running Spark on cluster
Cluster Managers (Schedulers)

- Spark's own Standalone cluster manager
- Hadoop YARN
- Apache Mesos

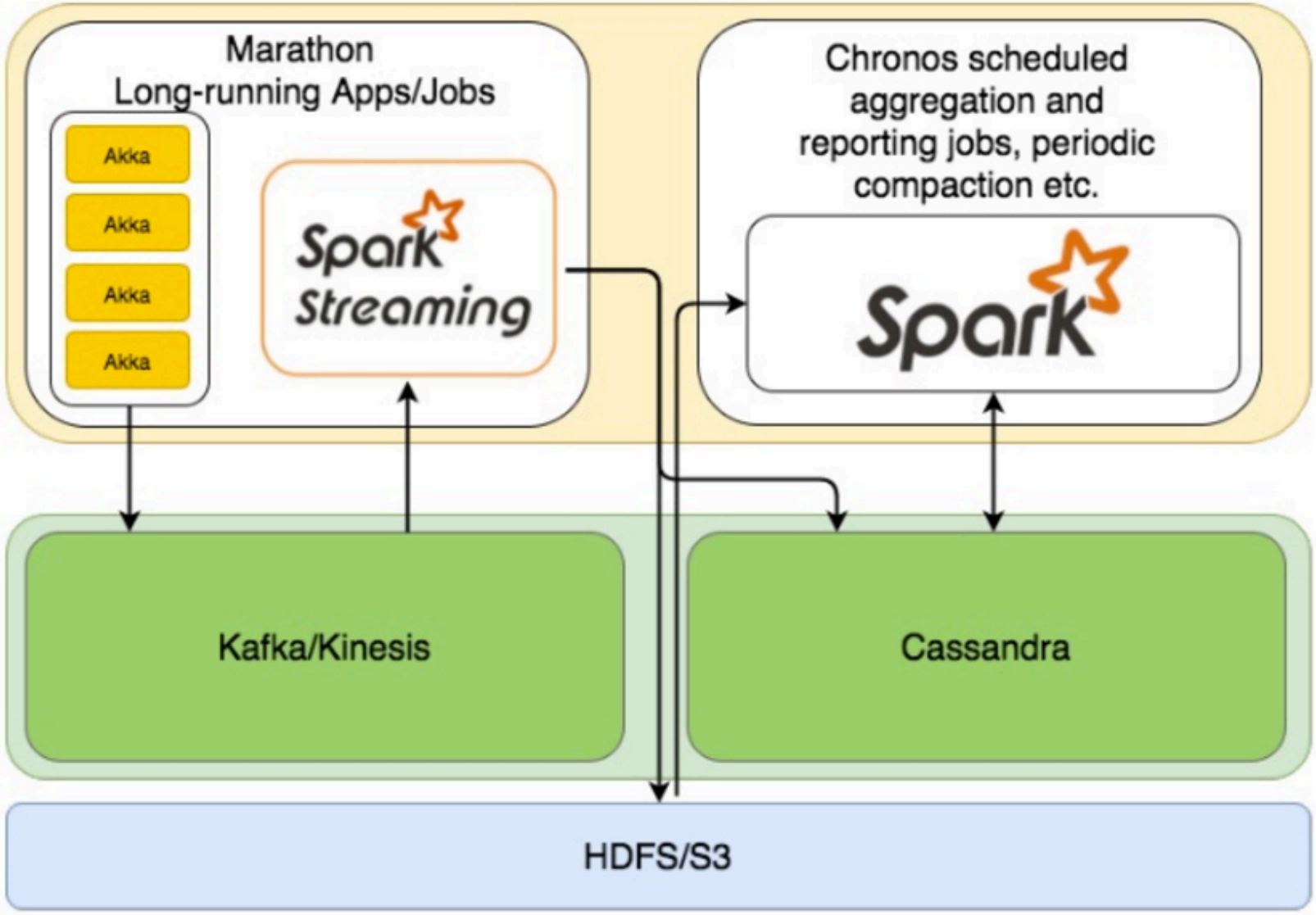
Source: https://jaceklaskowski.gitbooks.io/mastering-apache-spark/content/spark-cluster.html

Mesos Architecture Overview



Source: Anton Kirillov (2015), Data processing platforms architectures with Spark, Mesos, Akka, Cassandra and Kafka,

SMACK



Source: Anton Kirillov (2015), Data processing platforms architectures with Spark, Mesos, Akka, Cassandra and Kafka,

Spark Developer Resources

databricks PRODUCT SPARK SOLUTIONS CUSTOMERS COMPANY BLOG RESOURCES Partners Training Sign Up

Spark Developer Resources

Databricks provides a number of free resources online for Spark training, including course materials, video archives, sample apps, knowledge base, etc.

Note that course materials for these workshops are provided online under a Creative Commons Attribution-NonCommercial-NoDerivatives 4.0 International (CC BY-NC-ND 4.0) license.

Content Archives

Highlights of recent blogs, videos, and other community content:

- Apache Spark channel on YouTube (meetup livestream archives)
- Spark events worldwide
- Spark Packages
- O'Reilly Radar – Spark articles

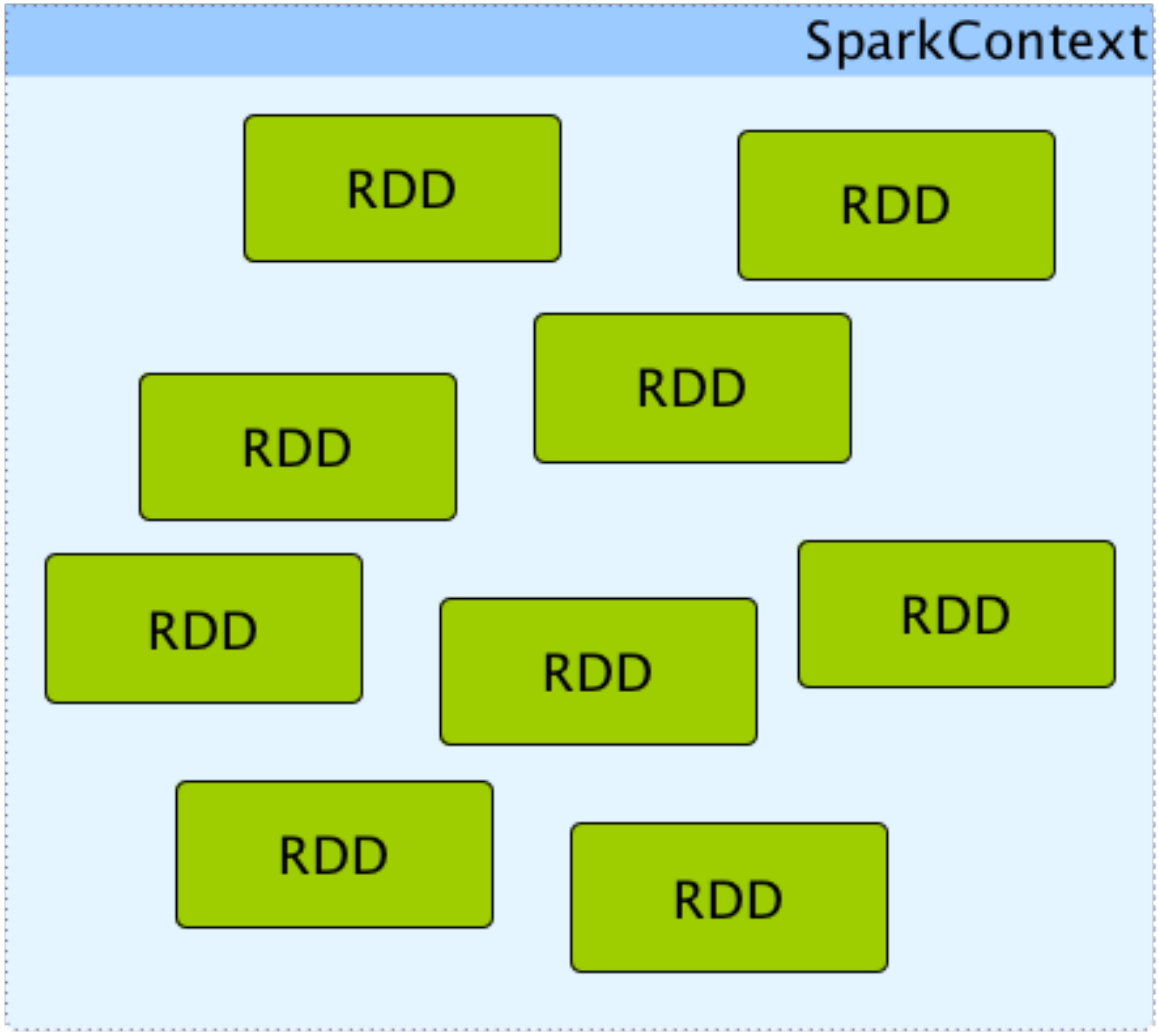
Spark ODBC Driver Download

Spark's ODBC driver lets you connect Business Intelligence (BI) and other third party applications to the Spark SQL server.

GET IT HERE

https://databricks.com/spark/developer-resources

SparkContext and RDDs



Source: https://jaceklaskowski.gitbooks.io/mastering-apache-spark/content/spark-sparkcontext.html

Resilient Distributed Dataset (RDD)

Resilient Distributed Datasets: A Fault-Tolerant Abstraction for In-Memory Cluster Computing

Matei Zaharia, Mosharaf Chowdhury, Tathagata Das, Ankur Dave, Justin Ma,
Murphy McCauley, Michael J. Franklin, Scott Shenker, Ion Stoica
University of California, Berkeley

Abstract

We present Resilient Distributed Datasets (RDDs), a distributed memory abstraction that lets programmers perform in-memory computations on large clusters in a fault-tolerant manner. RDDs are motivated by two types of applications that current computing frameworks handle inefficiently: iterative algorithms and interactive data mining tools. In both cases, keeping data in memory can improve performance by an order of magnitude. To achieve fault tolerance efficiently, RDDs provide a restricted form of shared memory, based on coarse-grained transformations rather than fine-grained updates to shared state. However, we show that RDDs are expressive enough to capture a wide class of computations, including recent specialized programming models for iterative jobs, such as Pregel, and new applications that these models do not capture. We have implemented RDDs in a system called Spark, which we evaluate through a variety of user applications and benchmarks.

1 Introduction

Cluster computing frameworks like MapReduce [10] and Dryad [19] have been widely adopted for large-scale data analytics. These systems let users write parallel computation, which can dominate application execution times.

Recognizing this problem, researchers have developed specialized frameworks for some applications that require data reuse. For example, Pregel [22] is a system for iterative graph computations that keeps intermediate data in memory, while HaLoop [7] offers an iterative MapReduce interface. However, these frameworks only support specific computation patterns (*e.g.*, looping a series of MapReduce steps), and perform data sharing implicitly for these patterns. They do not provide abstractions for more general reuse, *e.g.*, to let a user load several datasets into memory and run ad-hoc queries across them.

In this paper, we propose a new abstraction called *resilient distributed datasets (RDDs)* that enables efficient data reuse in a broad range of applications. RDDs are fault-tolerant, parallel data structures that let users explicitly persist intermediate results in memory, control their partitioning to optimize data placement, and manipulate them using a rich set of operators.

The main challenge in designing RDDs is defining a programming interface that can provide fault tolerance *efficiently*. Existing abstractions for in-memory storage on clusters, such as distributed shared memory [24], key-value stores [25], databases, and Piccolo [27], offer an

Source: https://www.cs.berkeley.edu/~matei/papers/2012/nsdi_spark.pdf

RDD

- **R**esilient
  - –fault-tolerant and so able to recompute missing or damaged partitions on node failures with the help of RDD lineage graph.
- **D**istributed across clusters.
- **D**ataset
  - –a collection of partitioned data.

Source: https://jaceklaskowski.gitbooks.io/mastering-apache-spark/content/spark-rdd.html

Resilient Distributed Datasets (RDD)

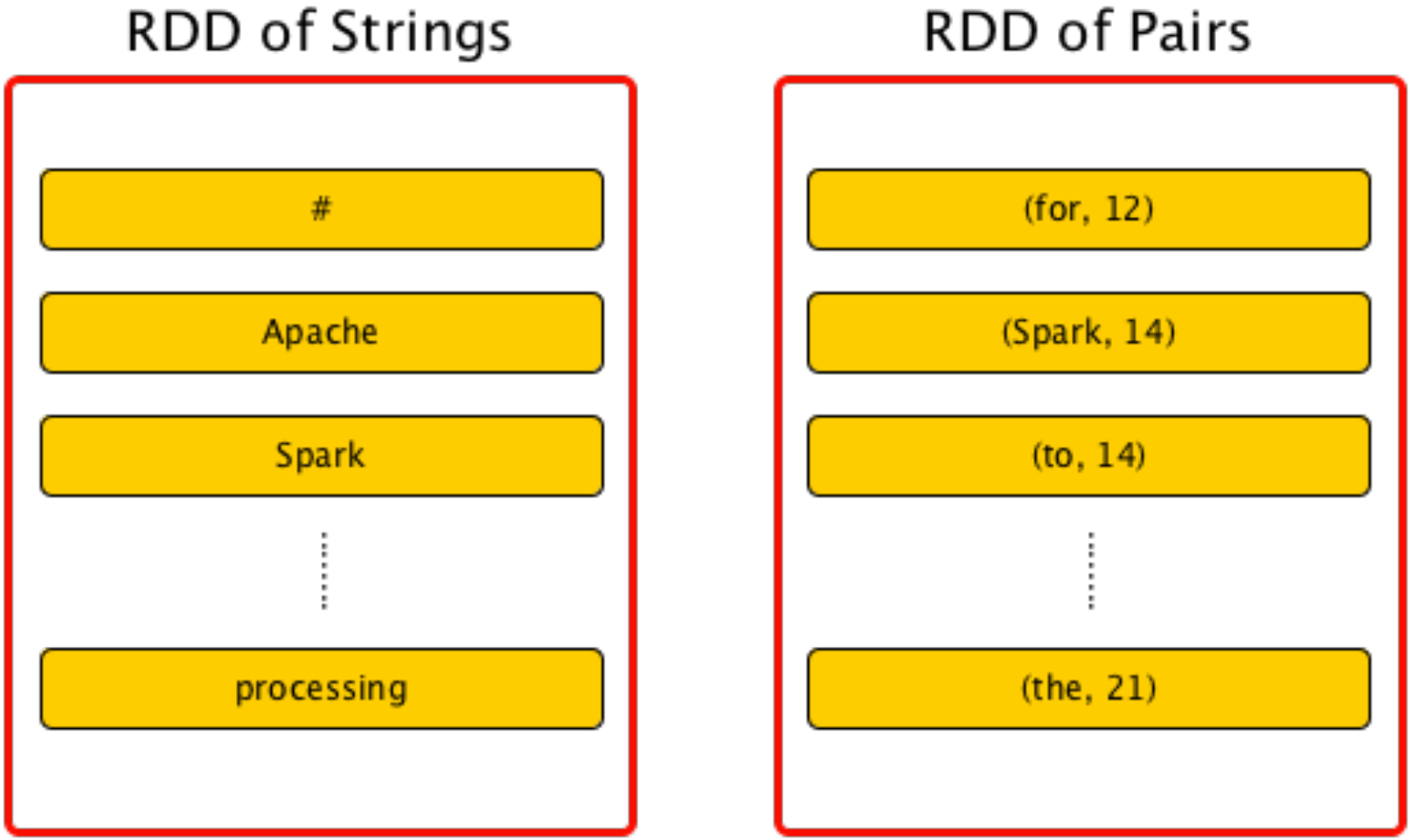
are the primary abstraction in Spark – a fault-tolerant collection of elements that can be operated on in parallel

Source: Databricks (2016), Intro to Apache Spark

Resilient Distributed Dataset (RDD)

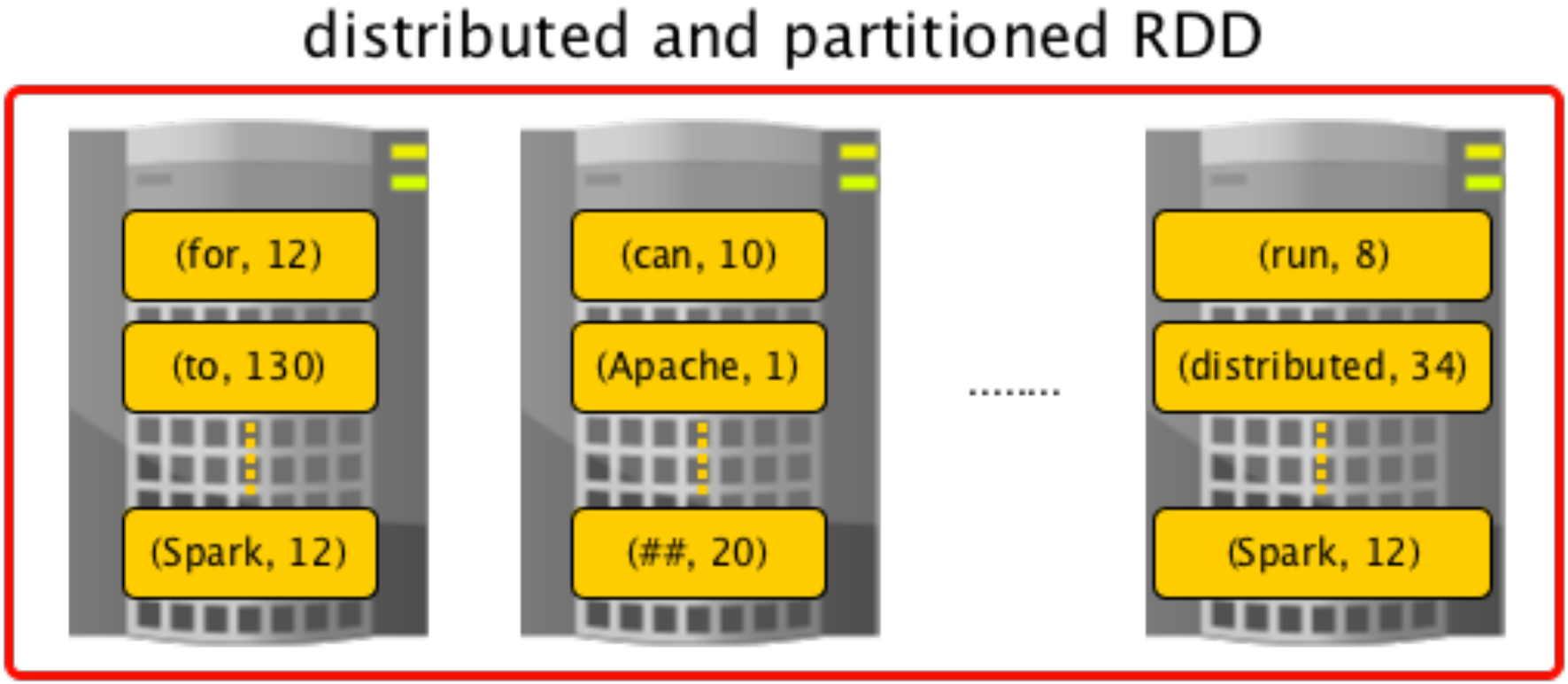
Spark's "Interface" to data

Spark RDD



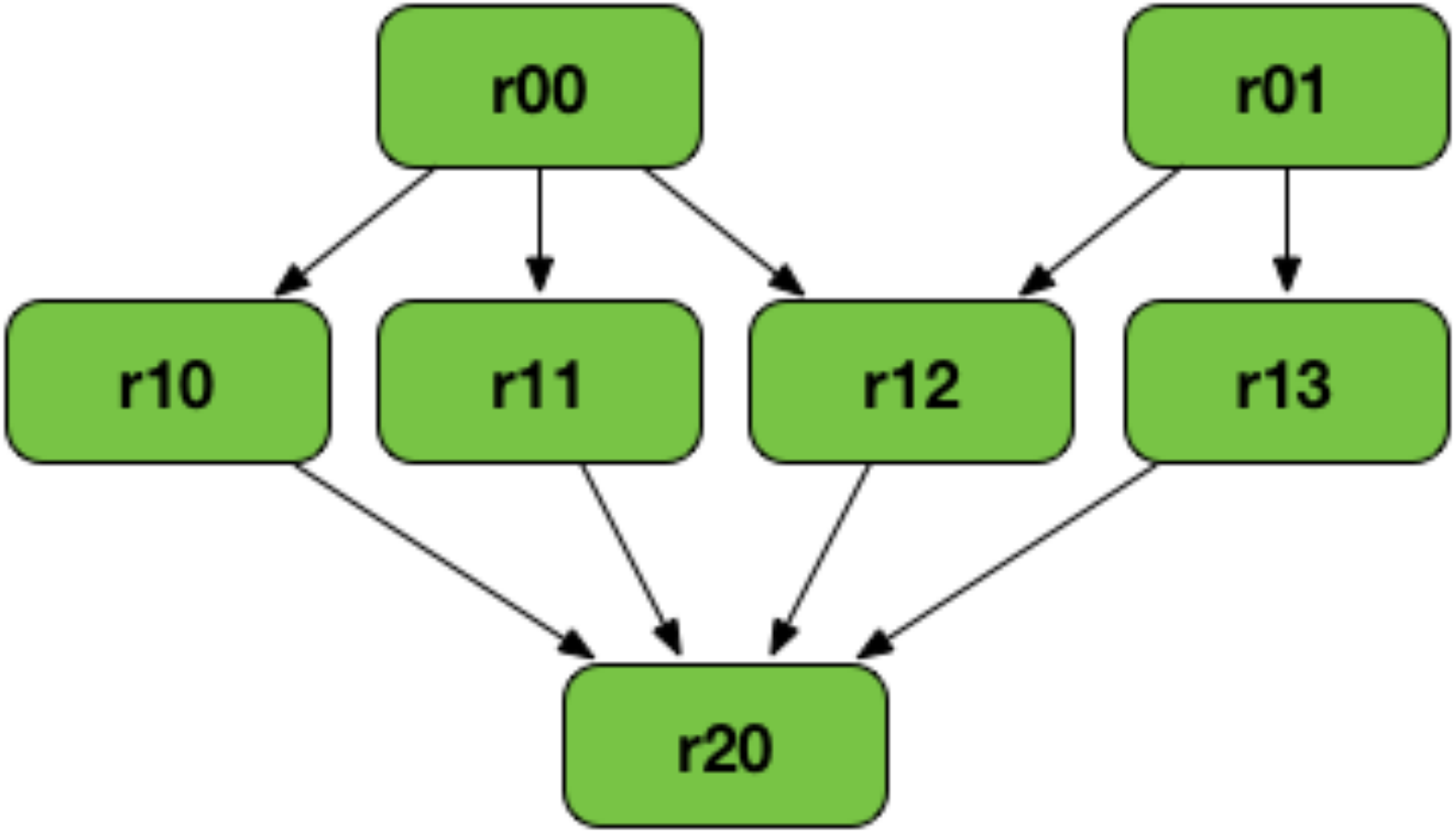
Source: https://jaceklaskowski.gitbooks.io/mastering-apache-spark/content/spark-rdd.html

Spark RDD



Source: https://jaceklaskowski.gitbooks.io/mastering-apache-spark/content/spark-rdd.html

RDD Lineage Graph (RDD operator graph)



Source: https://jaceklaskowski.gitbooks.io/mastering-apache-spark/content/spark-rdd.html

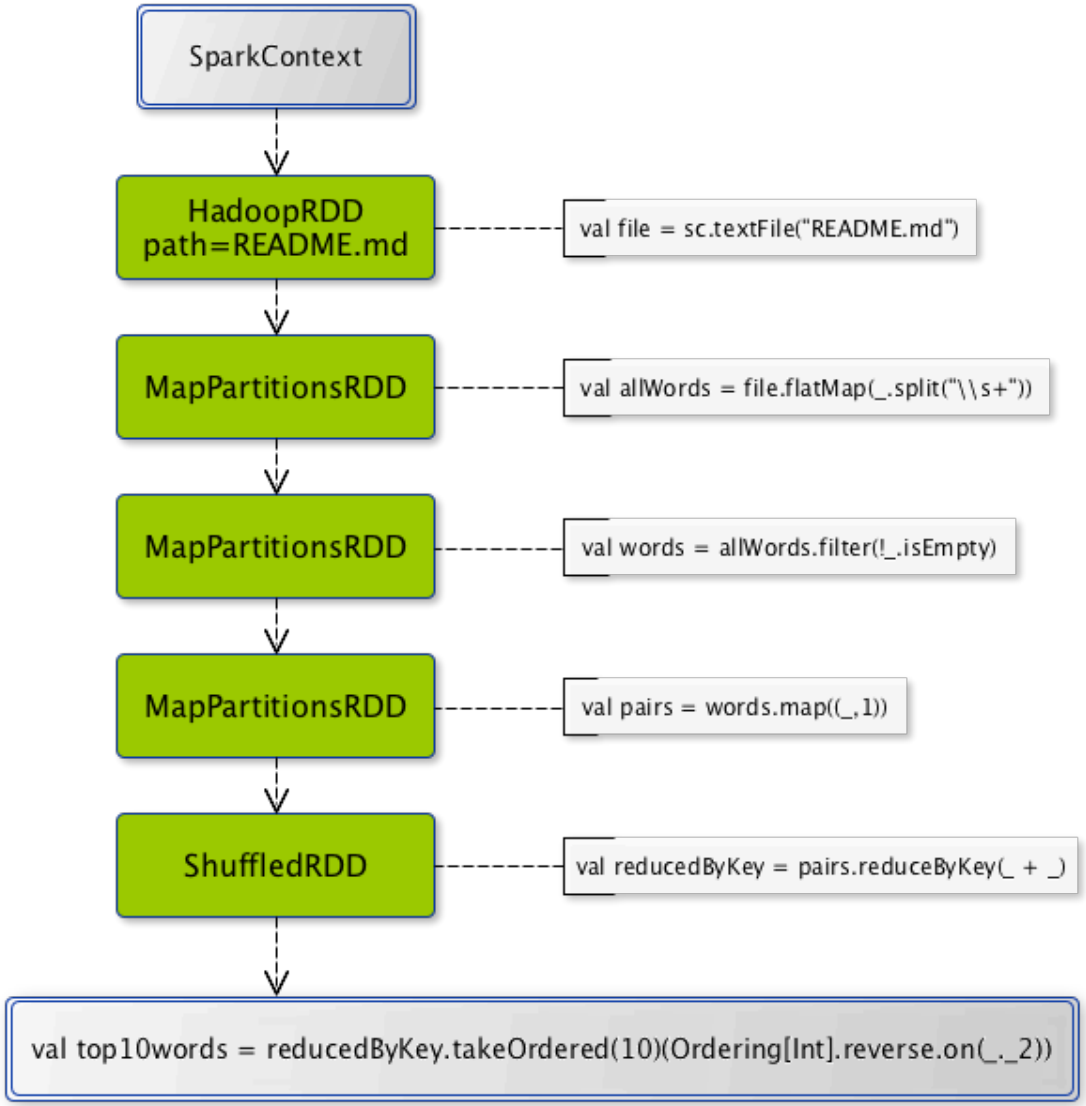
Resilient Distributed Dataset (RDD) Operations

Transformations

Actions

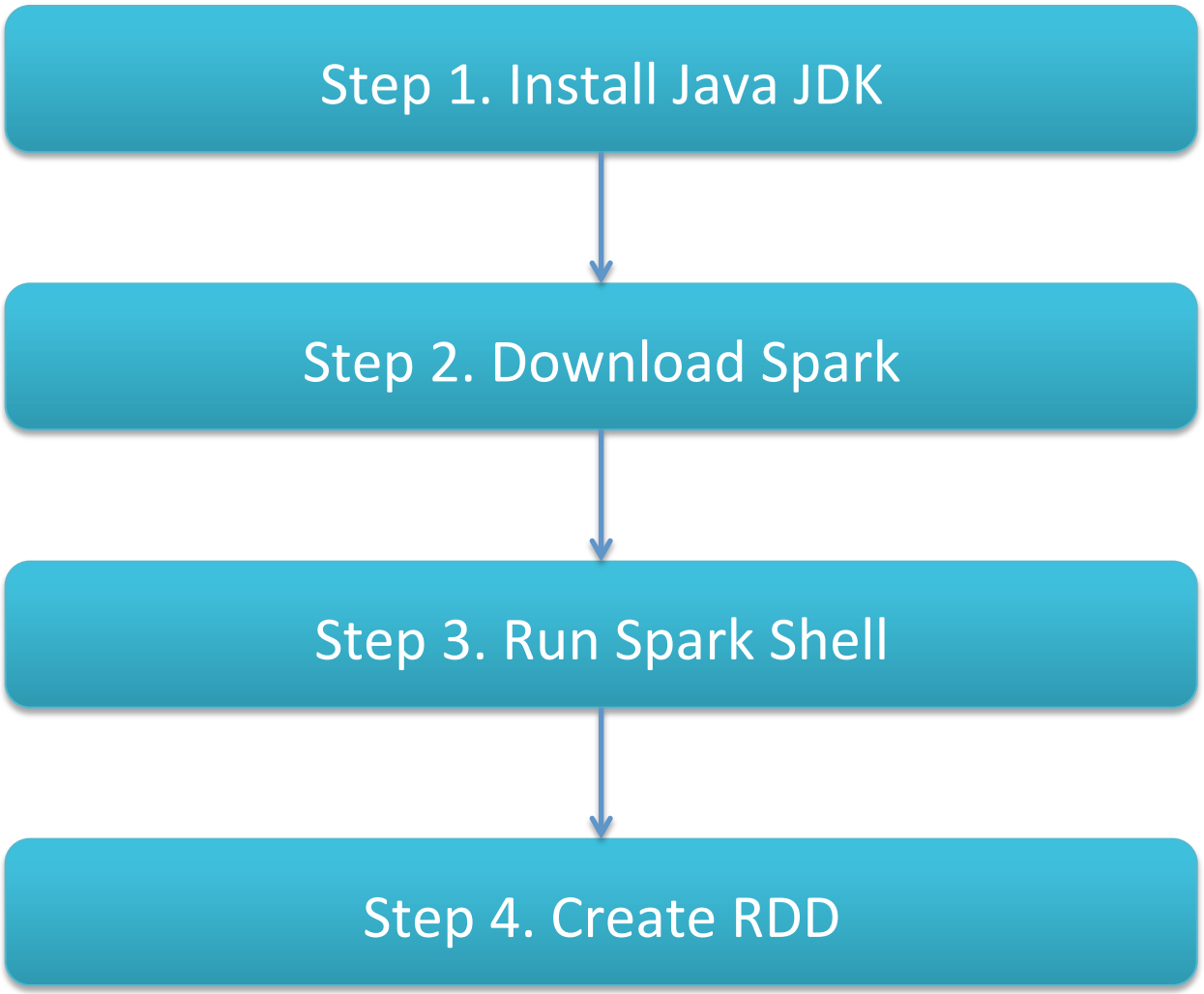
Source: Liondatasystems (2015), Intro to Apache Spark (Brain-Friendly Tutorial)

Spark RDD Operations



Source: https://jaceklaskowski.gitbooks.io/mastering-apache-spark/content/spark-rdd-operations.html

Get Started using Apache Spark on a Personal Computer (Windows/Mac OS X)



Source: Databricks (2016), Intro to Apache Spark

Apache Spark

spark.apache.org

Spark *Lightning-fast cluster computing*

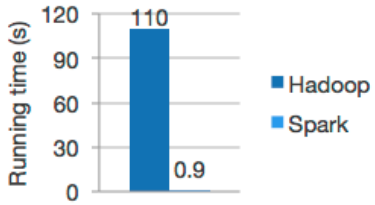
Download | Libraries | Documentation | Examples | Community | FAQ | Apache Software Foundation

Apache Spark™ is a fast and general engine for large-scale data processing.

Speed

Run programs up to 100x faster than Hadoop MapReduce in memory, or 10x faster on disk.

Spark has an advanced DAG execution engine that supports cyclic data flow and in-memory computing.



Logistic regression in Hadoop and Spark

Latest News

Submission is open for Spark Summit San Francisco (Feb 11, 2016)

Spark Summit East (Feb 16, 2016, New York) agenda posted (Jan 14, 2016)

Spark 1.6.0 released (Jan 04, 2016)

CFP for Spark Summit East 2016 is closing soon! (Nov 19, 2015)

Archive

Download Spark

Ease of Use

Write applications quickly in Java, Scala, Python, R.

Spark offers over 80 high-level operators that make it easy to build parallel apps. And you can use it *interactively* from the Scala, Python and R shells.

```
text_file = spark.textFile("hdfs://...")

text_file.flatMap(lambda line: line.split())
    .map(lambda word: (word, 1))
    .reduceByKey(lambda a, b: a+b)
```

Word count in Spark's Python API

Built-in Libraries:

SQL and DataFrames
Spark Streaming
MLlib (machine learning)
GraphX (graph)

Third-Party Packages

http://spark.apache.org/

Spark Download

Spark *Lightning-fast cluster computing*

Download | Libraries | Documentation | Examples | Community | FAQ | Apache Software Foundation

Download Apache Spark™

Our latest version is Spark 1.6.0, released on January 4, 2016 (release notes) (git tag)

1. Choose a Spark release: 1.6.0 (Jan 04 2016)
2. Choose a package type: Pre-built for Hadoop 2.6 and later
3. Choose a download type: Select Apache Mirror
4. Download Spark: spark-1.6.0-bin-hadoop2.6.tgz
5. Verify this release using the 1.6.0 signatures and checksums.

Note: Scala 2.11 users should download the Spark source package and build with Scala 2.11 support.

Link with Spark

Spark artifacts are hosted in Maven Central. You can add a Maven dependency with the following coordinates:

```
groupId: org.apache.spark
artifactId: spark-core_2.10
version: 1.6.0
```

Spark Source Code Management

If you are interested in working with the newest under-development code or contributing to Apache Spark development, you can also check out the master branch from Git:

Latest News

Submission is open for Spark Summit San Francisco (Feb 11, 2016)

Spark Summit East (Feb 16, 2016, New York) agenda posted (Jan 14, 2016)

Spark 1.6.0 released (Jan 04, 2016)

CFP for Spark Summit East 2016 is closing soon! (Nov 19, 2015)

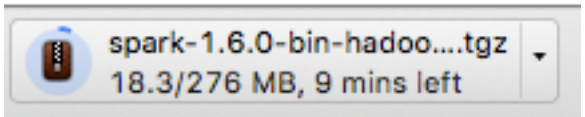
Archive

Download Spark

Built-in Libraries:

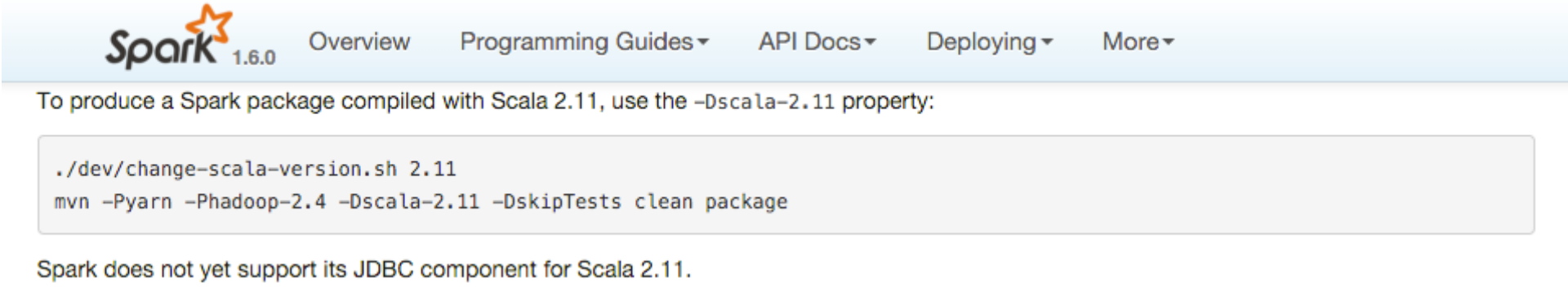
SQL and DataFrames
Spark Streaming
MLlib (machine learning)
GraphX (graph)

Third-Party Packages



http://spark.apache.org/downloads.html

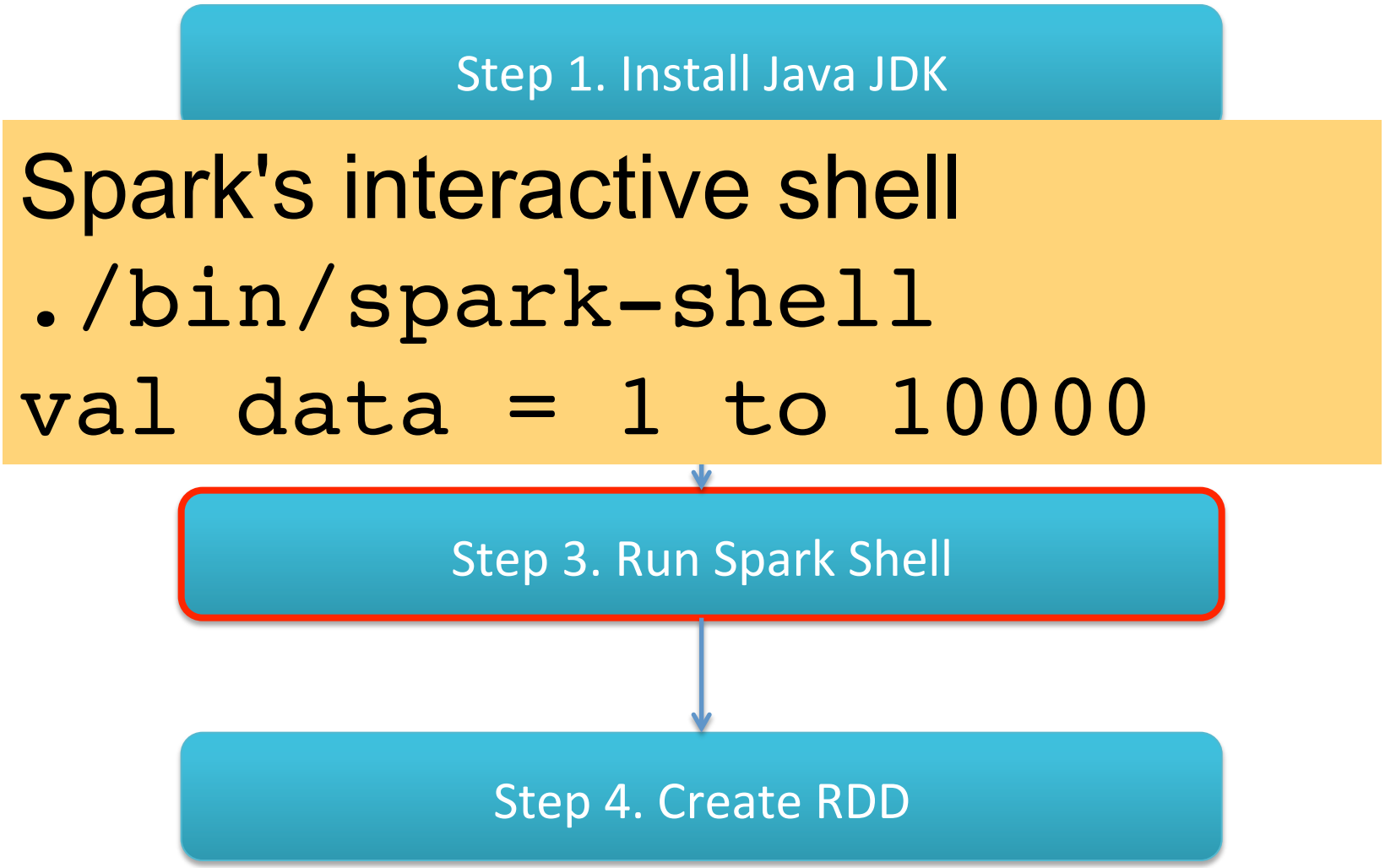
Scala 2.11




```
./dev/change-scala-version.sh 2.11
mvn -Pyarn -Phadoop-2.4 -Dscala-2.11 -DskipTests clean package
```

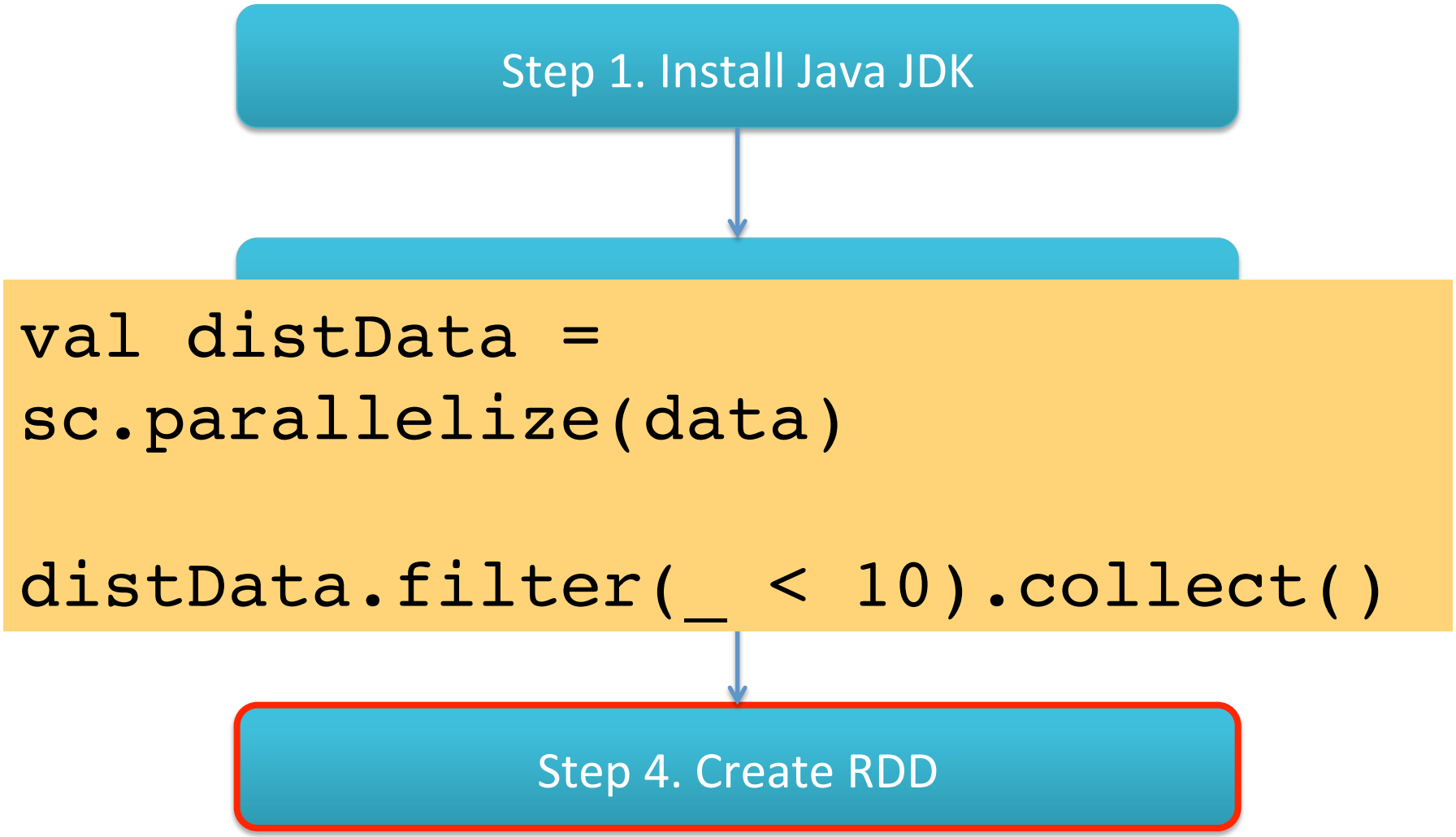
Source: http://spark.apache.org/docs/latest/building-spark.html#building-for-scala-211

Get Started using Apache Spark on a Personal Computer (Windows/Mac OS X)



Source: Databricks (2016), Intro to Apache Spark

Get Started using Apache Spark on a Personal Computer (Windows/Mac OS X)



Source: Databricks (2016), Intro to Apache Spark

Word Count

Hello World in Big Data

Spark Examples

Lightning-fast cluster computing

Download | Libraries | Documentation | Examples | Community | FAQ | Apache Software Foundation

Spark Examples

These examples give a quick overview of the Spark API. Spark is built on the concept of *distributed datasets*, which contain arbitrary Java or Python objects. You create a dataset from external data, then apply parallel operations to it. The building block of the Spark API is its RDD API. In the RDD API, there are two types of operations: *transformations*, which define a new dataset based on previous ones, and *actions*, which kick off a job to execute on a cluster. On top of Spark's RDD API, high level APIs are provided, e.g. DataFrame API and Machine Learning API. These high level APIs provide a concise way to conduct certain data operations. In this page, we will show examples using RDD API as well as examples using high level APIs.

RDD API Examples

Word Count

In this example, we use a few transformations to build a dataset of (String, Int) pairs called `counts` and then save it to a file.

Python | Scala | Java

```
text_file = sc.textFile("hdfs://...")
counts = text_file.flatMap(lambda line: line.split(" ")) \
             .map(lambda word: (word, 1)) \
             .reduceByKey(lambda a, b: a + b)
counts.saveAsTextFile("hdfs://...")
```

Latest News

Submission is open for Spark Summit San Francisco (Feb 11, 2016)

Spark Summit East (Feb 16, 2016, New York) agenda posted (Jan 14, 2016)

Spark 1.6.0 released (Jan 04, 2016)

CFP for Spark Summit East 2016 is closing soon! (Nov 19, 2015)

Archive

Download Spark

Built-in Libraries:

SQL and DataFrames
Spark Streaming
MLlib (machine learning)
GraphX (graph)

Third-Party Packages

Source: http://spark.apache.org/examples.html

Spark Word Count Python

```
text_file = sc.textFile("hdfs://...")
counts = text_file.flatMap(lambda line: line.split(" ")) \
             .map(lambda word: (word, 1)) \
             .reduceByKey(lambda a, b: a + b)
counts.saveAsTextFile("hdfs://...")
```

Source: http://spark.apache.org/examples.html

Spark Word Count Scala

```
val textFile = sc.textFile("hdfs://...")
val counts = textFile.flatMap(line => line.split(" "))
                 .map(word => (word, 1))
                 .reduceByKey(_ + _)
counts.saveAsTextFile("hdfs://...")
```

Source: http://spark.apache.org/examples.html

Spark Word Count Java

```
JavaRDD<String> textFile = sc.textFile("hdfs://...");
JavaRDD<String> words = textFile.flatMap(new FlatMapFunction<String, String>()
{
  public Iterable<String> call(String s) { return Arrays.asList(s.split("
")); }
});
JavaPairRDD<String, Integer> pairs = words.mapToPair(new PairFunction<String,
String, Integer>() {
  public Tuple2<String, Integer> call(String s) { return new Tuple2<String,
Integer>(s, 1); }
});
JavaPairRDD<String, Integer> counts = pairs.reduceByKey(new Function2<Integer,
Integer, Integer>() {
  public Integer call(Integer a, Integer b) { return a + b; }
});
counts.saveAsTextFile("hdfs://...");
```

Source: http://spark.apache.org/examples.html

Spark Word Count

Python

```
text_file = sc.textFile("hdfs://...")
counts = text_file.flatMap(lambda line: line.split(" ")) \
             .map(lambda word: (word, 1)) \
             .reduceByKey(lambda a, b: a + b)
counts.saveAsTextFile("hdfs://...")
```

Scala

```
val textFile = sc.textFile("hdfs://...")
val counts = textFile.flatMap(line => line.split(" "))
                 .map(word => (word, 1))
                 .reduceByKey(_ + _)
counts.saveAsTextFile("hdfs://...")
```

Java

```
JavaRDD<String> textFile = sc.textFile("hdfs://...");
JavaRDD<String> words = textFile.flatMap(new FlatMapFunction<String, String>() {
  public Iterable<String> call(String s) { return Arrays.asList(s.split(" ")); }
});
JavaPairRDD<String, Integer> pairs = words.mapToPair(new PairFunction<String, String, Integer>() {
  public Tuple2<String, Integer> call(String s) { return new Tuple2<String, Integer>(s, 1); }
});
JavaPairRDD<String, Integer> counts = pairs.reduceByKey(new Function2<Integer, Integer, Integer>() {
  public Integer call(Integer a, Integer b) { return a + b; }
});
counts.saveAsTextFile("hdfs://...");
```

Source: http://spark.apache.org/examples.html

Spark Word Count Python

```
text_file = sc.textFile(”input_readme.txt")
counts = text_file.flatMap(lambda line: line.split(" "))
      .map(lambda word: (word, 1))
      .reduceByKey(lambda a, b: a + b)
counts.saveAsTextFile(”output_wordcount.txt")
```

Source: http://spark.apache.org/examples.html

WordCount Example

- Input

```
Hello World Bye World
Hello Hadoop Goodbye Hadoop
```

- For the given sample input the map emits

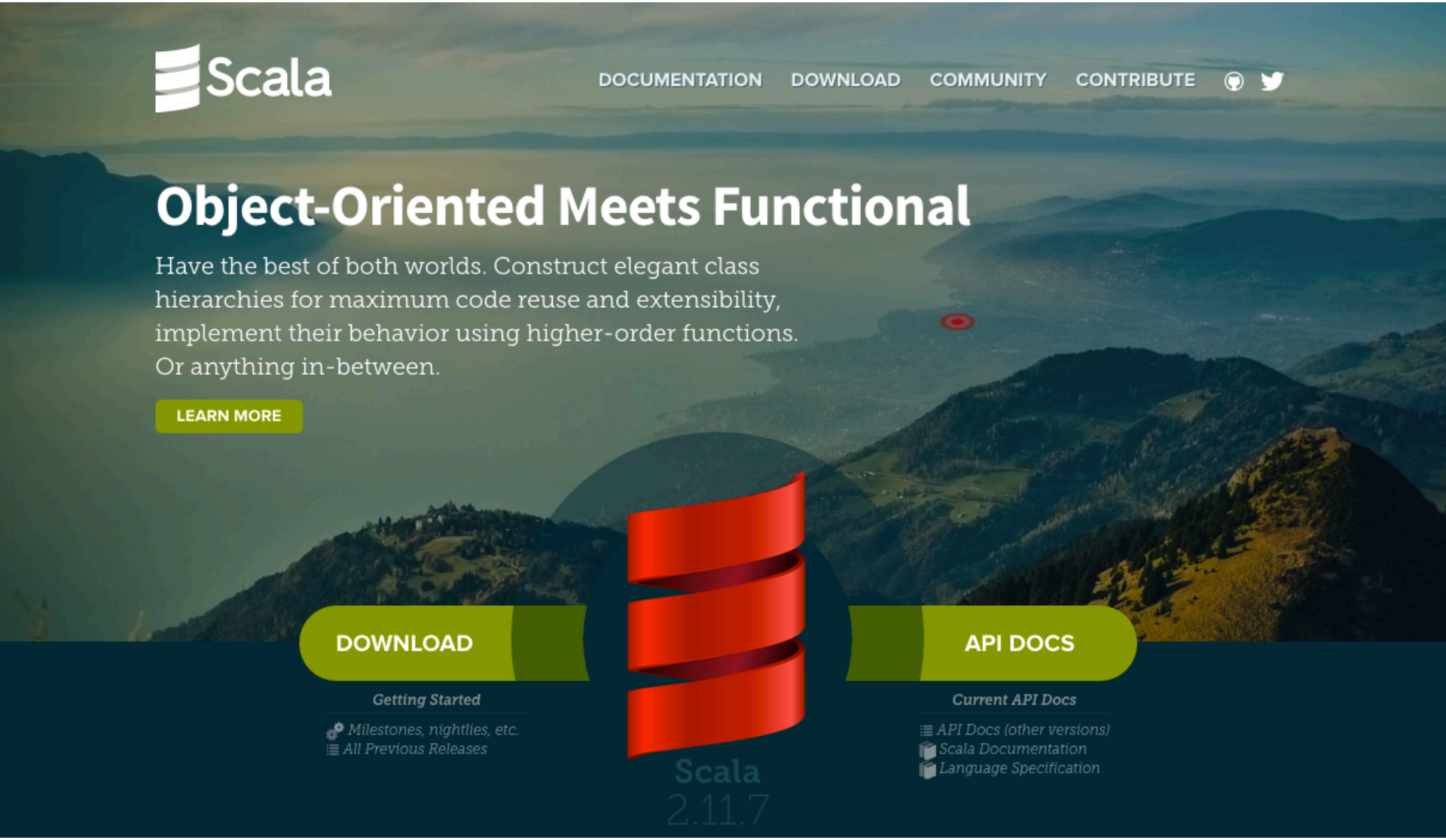
```
< Hello, 1>
< World, 1>
< Bye, 1>
< World, 1>
< Hello, 1>
< Hadoop, 1>
< Goodbye, 1>
< Hadoop, 1>
```

-

```
< Bye, 1>
< Goodbye, 1>
< Hadoop, 2>
< Hello, 2>
< World, 2>
```

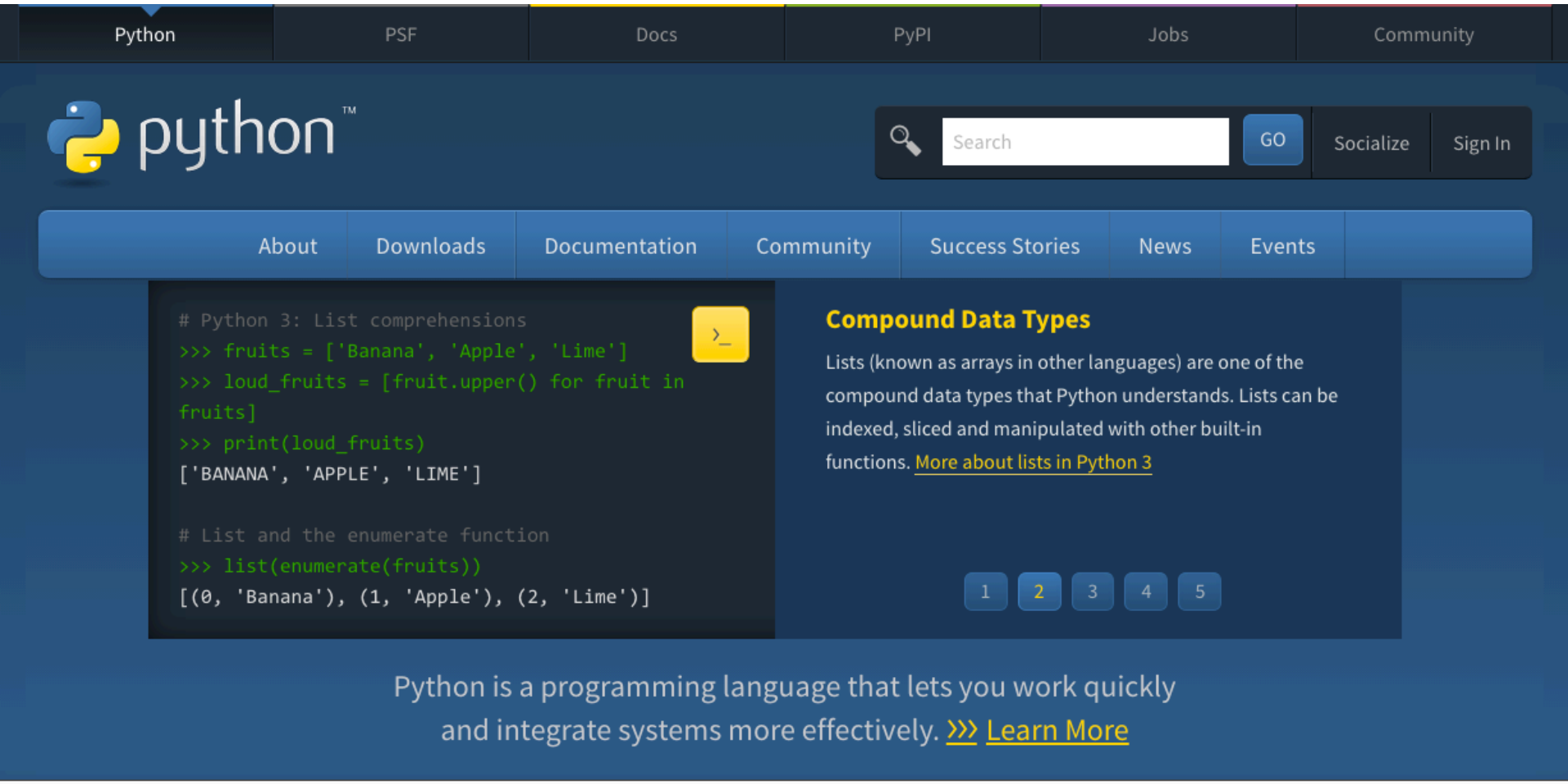
Source: http://www.slideshare.net/tcloudcomputing-tw/hadoop-product-family-2013q3-t-cloud-computing-26218439

Scala



http://www.scala-lang.org/

Python



https://www.python.org/

PySpark: Spark Python API

PySpark 1.6.0 documentation »

Table Of Contents

Welcome to Spark Python API Docs!
- Core classes:

Indices and tables

Next topic

pyspark package

This Page

Show Source

Quick search

Go

Enter search terms or a module, class or function name.

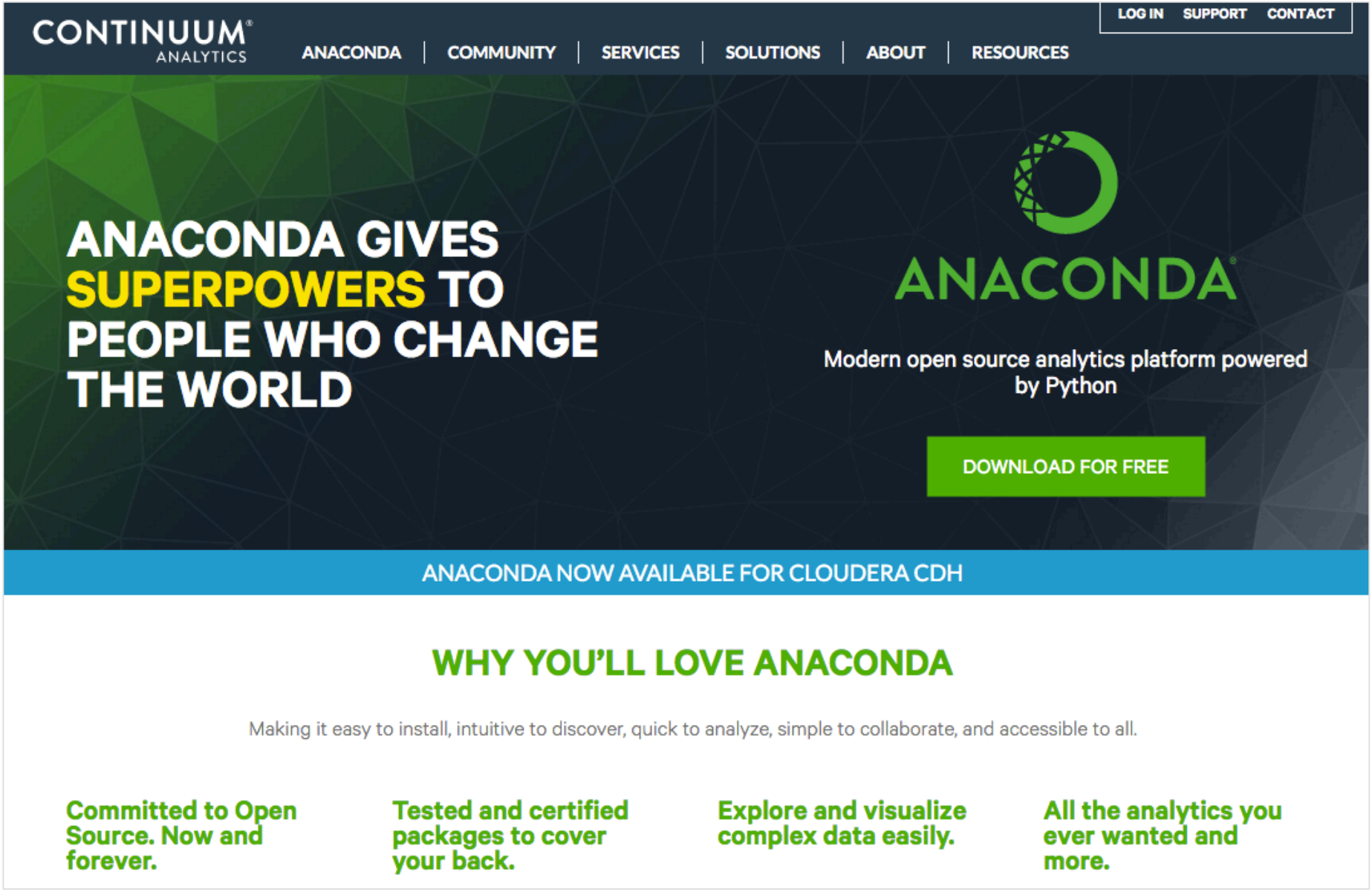
Welcome to Spark Python API Docs!

Contents:

- pyspark package
  - Subpackages
  - Contents
- pyspark.sql module
  - Module Context
  - pyspark.sql.types module
  - pyspark.sql.functions module
- pyspark.streaming module
  - Module contents
  - pyspark.streaming.kafka module
  - pyspark.streaming.kinesis module
  - pyspark.streaming.flume.module
  - pyspark.streaming.mqtt module
- pyspark.ml package
  - ML Pipeline APIs
  - pyspark.ml.param module
  - pyspark.ml.feature module
  - pyspark.ml.classification module
  - pyspark.ml.clustering module
  - pyspark.ml.recommendation module
  - pyspark.ml.regression module
  - pyspark.ml.tuning module
  - pyspark.ml.evaluation module
- pyspark.mllib package
  - pyspark.mllib.classification module
  - pyspark.mllib.clustering module
  - pyspark.mllib.evaluation module
  - pyspark.mllib.feature module

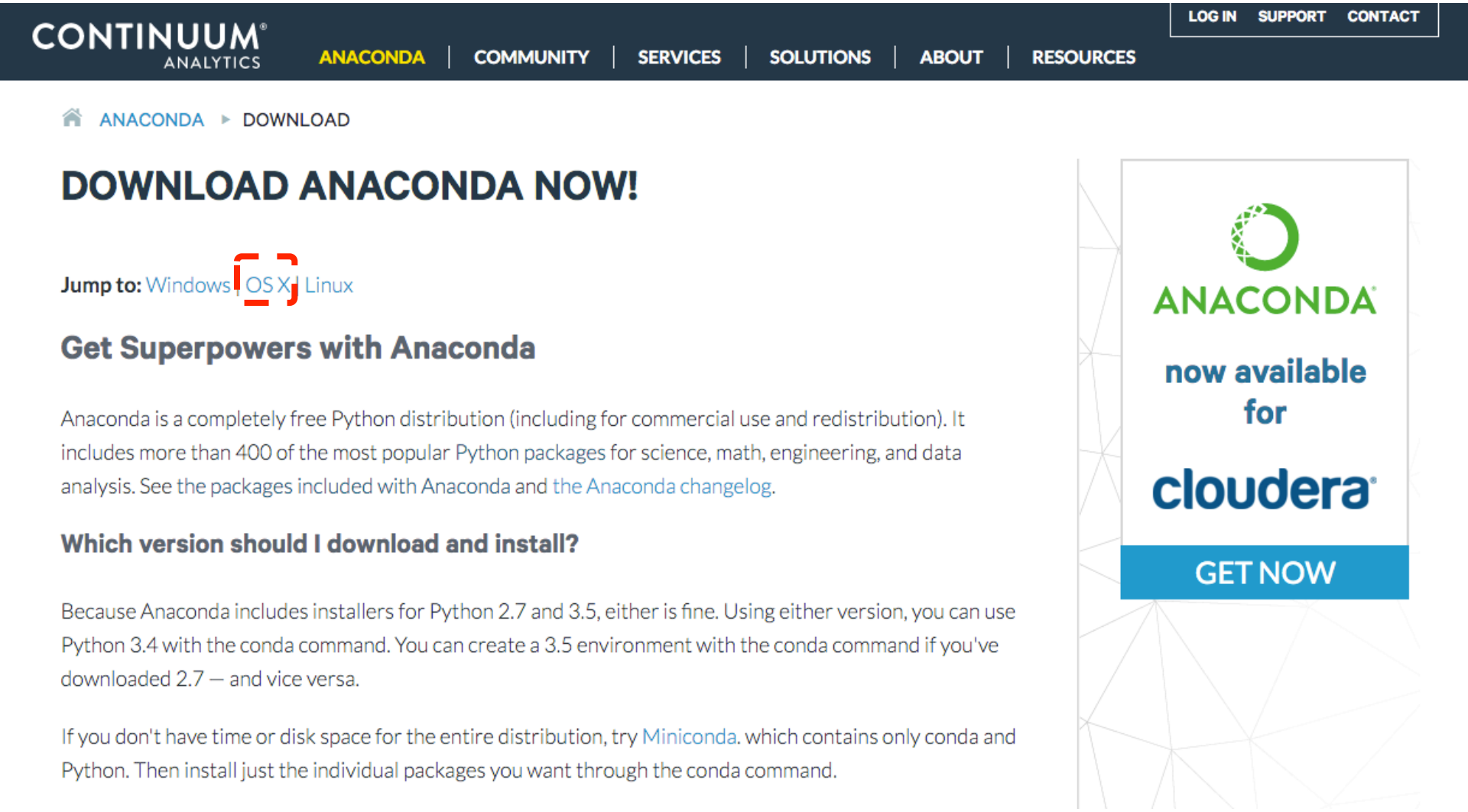
http://spark.apache.org/docs/latest/api/python/

Anaconda



https://www.continuum.io/

Download Anaconda



https://www.continuum.io/downloads

Download Anaconda Python 2.7

Anaconda for OS X

OS X Anaconda Installation

Choose either the graphical installer or the command line installer for OS X.

Graphical Installer:

1. Download the graphical installer.
2. Double-click the downloaded .pkg file and follow the instructions.

https://www.continuum.io/downloads

Anadonda for Cloudera CDH

Modern Open Source Analytics Powered by Python

Anaconda empowers the entire Data Science team – Data Scientists, Data Engineers, Developers, DevOps, Data Engineers, and Business Analysts – to analyze data in Hadoop and deliver high value, high impact predictive and machine learning solutions with Python.

SEE PACKAGES IN ANACONDA | GET STARTED WITH CLOUDERA CDH | READ ANACONDA E

Leave us a message

http://know.continuum.io/anaconda-for-cloudera.html

Anadonda for Cloudera CDH

Make Python On Hadoop Easy As 1-2-3

- Install Anaconda on a CDH cluster
- Build and run Python based solutions easily across a Cloudera CDH cluster and alongside PySpark jobs

Source: http://know.continuum.io/anaconda-for-cloudera.html

Anadonda for Cloudera CDH

Leverage Anaconda For Distributed Data Science On Cloudera

- Give your Data Scientists the most popular Python packages they know and love
- Empower your Data Science team to explore, build and deploy predictive models

Source: http://know.continuum.io/anaconda-for-cloudera.html

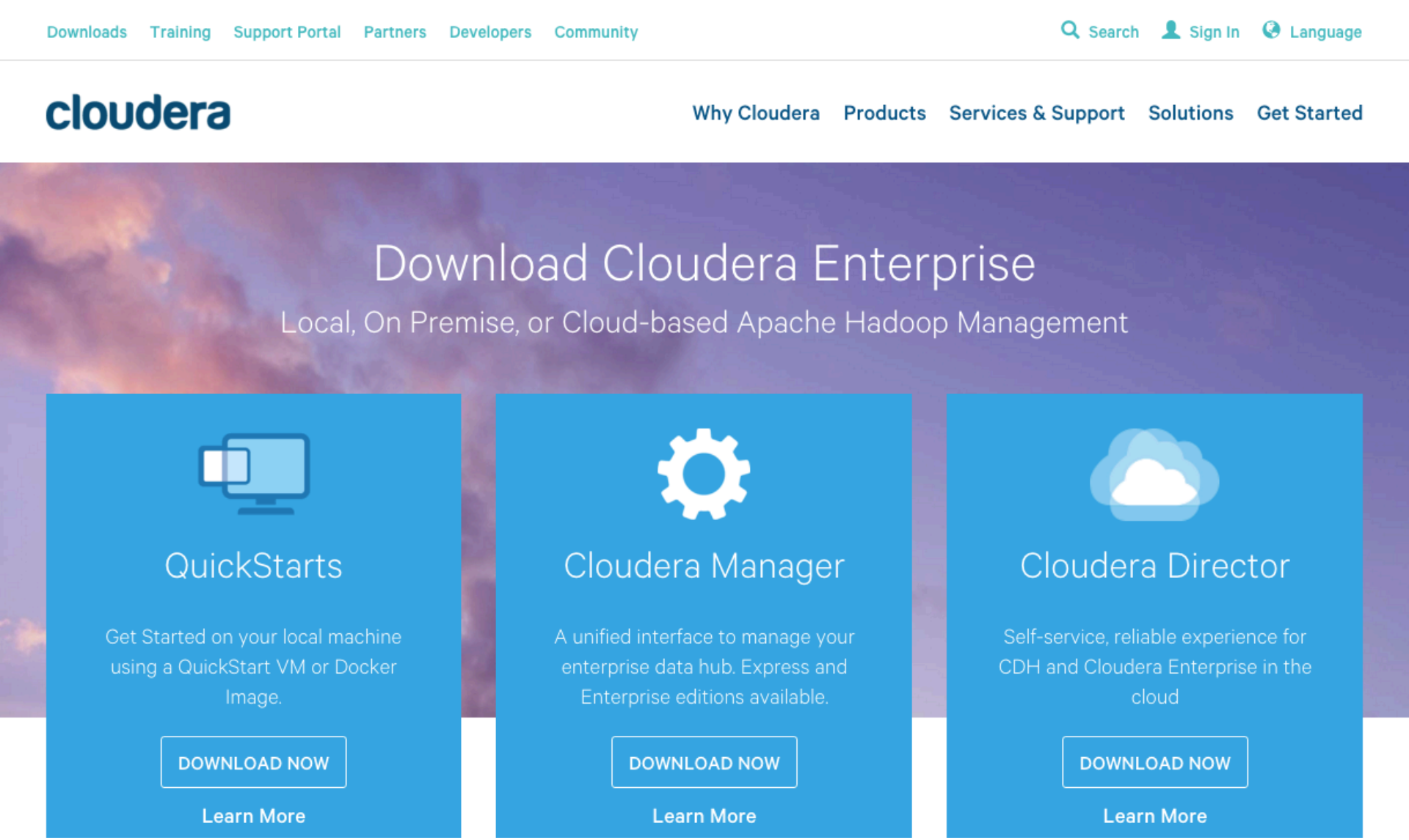
Anadonda for Cloudera CDH

Transition To Enterprise To Get Full Power Of Anaconda

- Dynamically manage multiple Python environments alongside a CDH cluster
- Get security, authentication and many other enterprise requirements with the paid Anaconda Workgroup and Anaconda Enterprise subscriptions
- Read our blog on Getting the Most Out of Anaconda for Your Cluster.

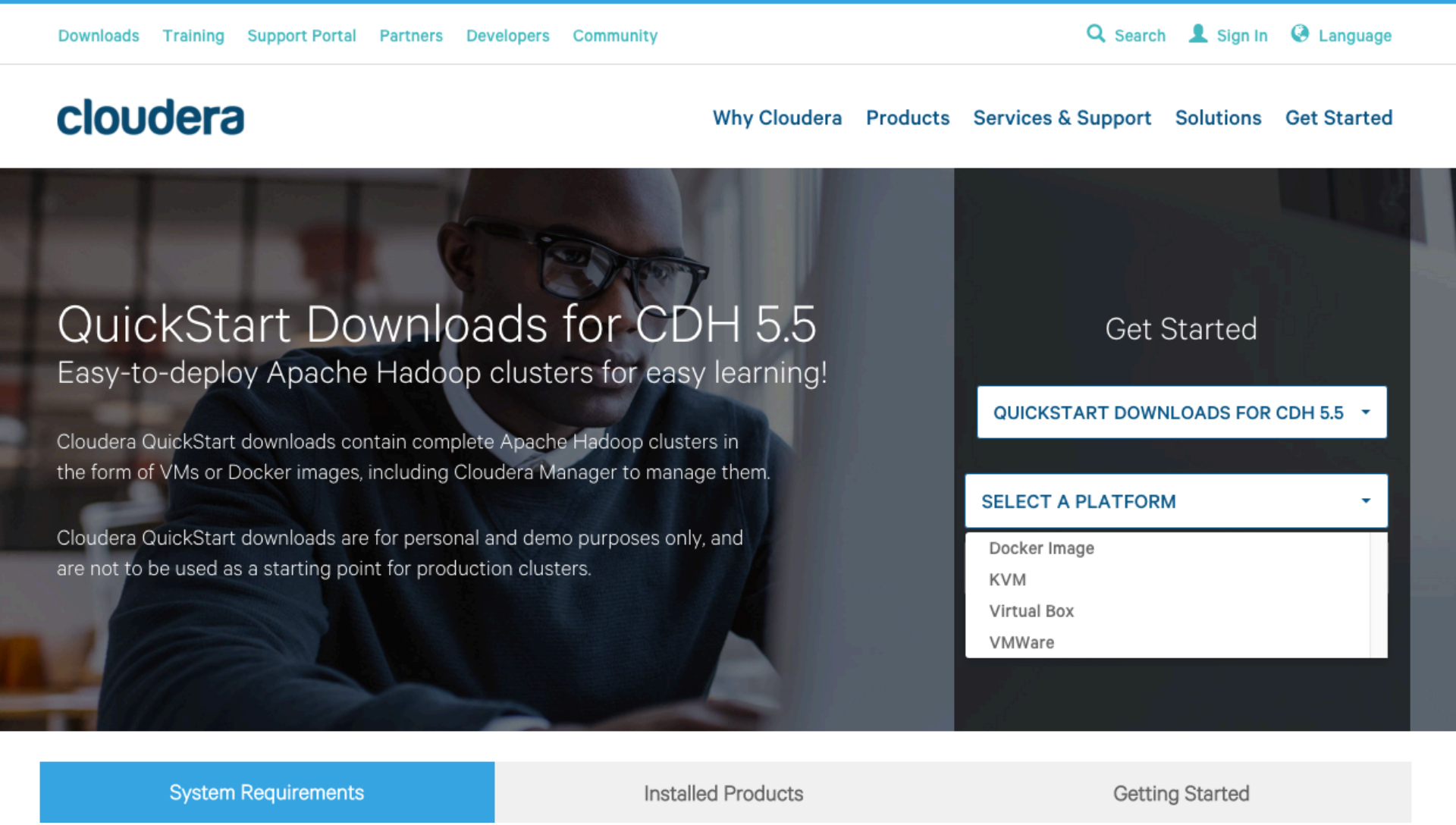
Source: http://know.continuum.io/anaconda-for-cloudera.html

Cloudera



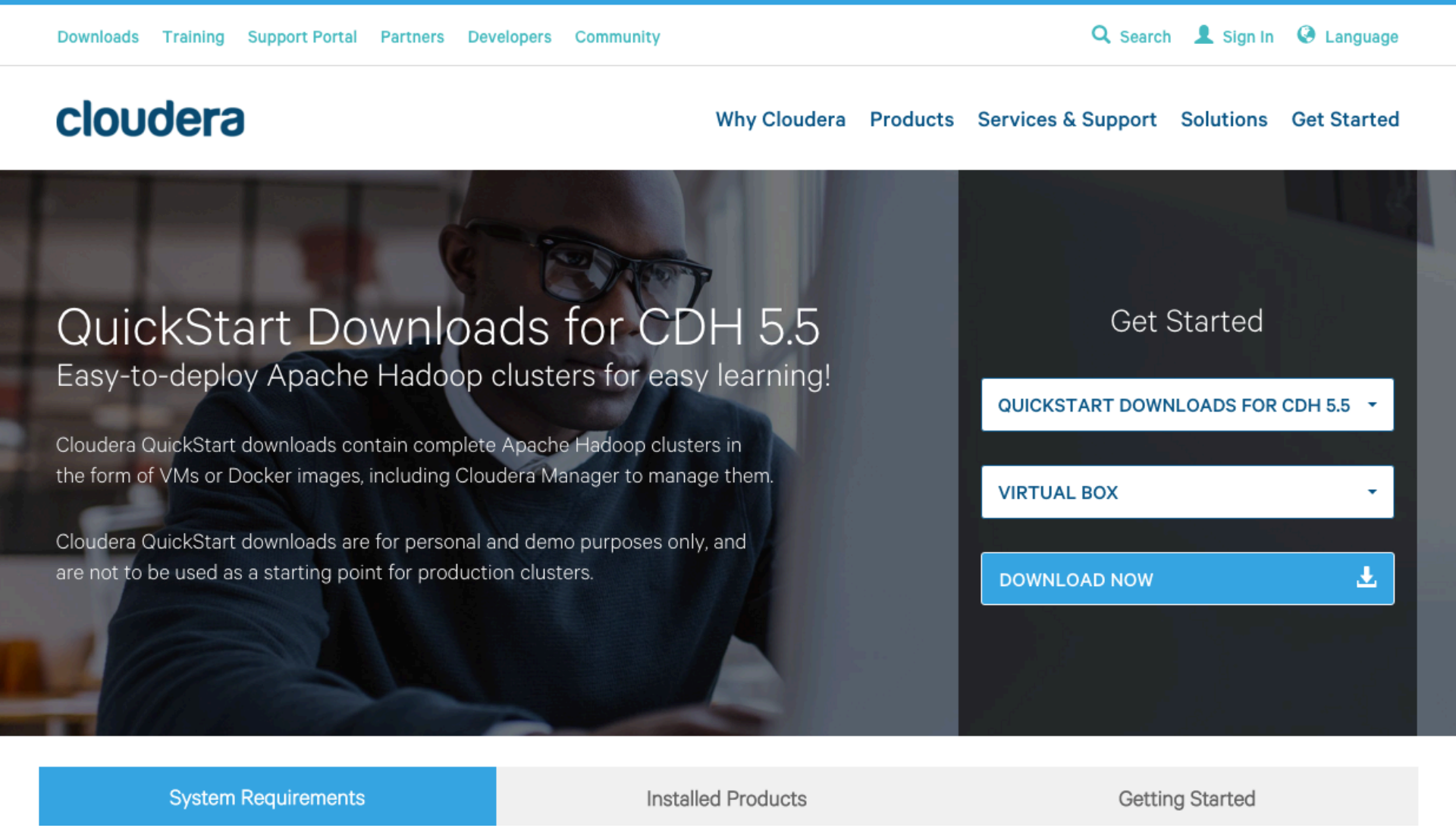
http://www.cloudera.com/downloads.html

Cloudera CDH 5.5



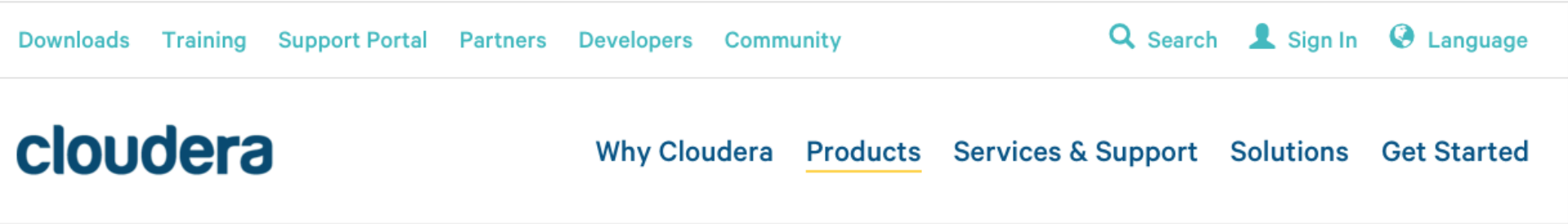
http://www.cloudera.com/downloads/quickstart_vms/5-5.html

Cloudera CDH 5.5



http://www.cloudera.com/downloads/quickstart_vms/5-5.html

Cloudera CDH



CDH Components

CDH is Cloudera's 100% open source platform distribution, including Apache Hadoop and built specifically to meet enterprise demands. CDH delivers everything you need for enterprise use right out of the box. By integrating Hadoop with more than a dozen other critical open source projects, Cloudera has created a functionally advanced system that helps you perform end-to-end Big Data workflows.

TRY CDH

http://www.cloudera.com/products/apache-hadoop/key-cdh-components.html

Spark Programming Guide

http://spark.apache.org/docs/latest/programming-guide.html

References

- Mike Frampton (2015),
  Mastering Apache Spark, Packt Publishing
- Anton Kirillov (2015), Data processing platforms architectures with Spark, Mesos, Akka, Cassandra and Kafka, Big Data AW Meetup,
  http://www.slideshare.net/akirillov/data-processing-platforms-architectures-with-spark-mesos-akka-cassandra-and-kafka
- Spark Apache (2016), PySpark: Spark Python API,
  http://spark.apache.org/docs/latest/api/python/
- Spark Apache (2016), Spark Programming Guide,
  http://spark.apache.org/docs/latest/programming-guide.html
- Spark Submit (2014), Spark Summit 2014 Training,
  https://spark-summit.org/2014/training
- Databricks (2016), Spark Developer Resources,
  https://databricks.com/spark/developer-resources
- Liondatasystems (2015), Intro to Apache Spark (Brain-Friendly Tutorial),
  https://www.youtube.com/watch?v=rvDpBTV89AM
- Databricks (2016), Intro to Apache Spark,
  http://training.databricks.com/workshop/itas_workshop.pdf