SPRINGONE2GX
WASHINGTON, DC

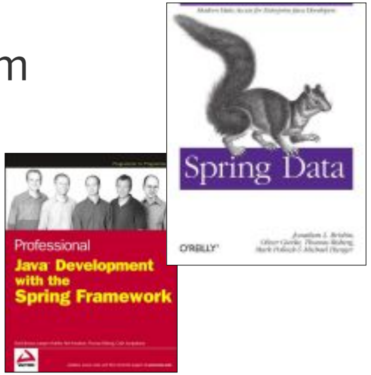
Hadoop Workflows using Spring Technologies

Thomas Risberg
@trisberg

Speaker

Thomas Risberg

- Member of the Spring Data engineering team at Pivotal
- Lead for the Spring for Apache Hadoop project
- Joined the Spring Framework open source project in 2003 working on JDBC support
- co-author of “Professional Java Development with Spring Framework” from Wrox 2005 and “Spring Data” book from O'Reilly 2012



Agenda

- Background and introduction to “Spring for Apache Hadoop”
- What’s new in “Spring for Apache Hadoop”
  - HiveServer2/JDBC replaces HiveServer1/Thrift Client support
  - Improved `@Configuration` support
  - Sqoop2 and Spark batch tasklet support
- Spring Boot and Batch examples:
  - Spring Boot and Data Ingestion
  - HiveServer2 batch job
  - Spark on YARN batch job
- Cloud native Hadoop data microservices
  - Programming model for Spring Cloud Streams

Spring for Apache Hadoop

“ *Spring for Apache Hadoop provides extensions to Spring, Spring Boot, Spring Batch, and Spring Integration to build manageable and robust pipeline solutions around Hadoop.*

Consistent Programming Model

- Configure, and parameterize Hadoop connectivity and all job types
- Support for running MapReduce jobs, streaming, tool, jars
- Configure Hadoop's distributed cache
- Support for working with Hive, Pig, HBase, Sqoop2, Spark and MapReduce
- Writing to HDFS – partitioning, many data formats
- Support for YARN programming
- Relies on standard Spring Framework features
  - Configuration and property files
  - Environment profiles – easily move application from dev to qa to prod

Developer Productivity

- Create well-formed applications, not spaghetti script applications
- Simplify HDFS access:
  - `FsShell` API with support for JVM scripting
  - Powerful and flexible `DataStoreWriter` implementations
- Helper “Template” classes for Hive/Pig/HBase
- Runner classes for Hive/Pig/MapReduce for small workflows
- Tasklet implementations for larger Spring Batch flows
  - Hive, Pig, Spark, Sqoop2, MapReduce

Common Use Cases

- Apply across a wide range of use cases
  - Ingestion: Events/JDBC/NoSQL/Files to HDFS
  - Orchestrate: Hadoop Jobs
  - Export: HDFS to JDBC/NoSQL
- Spring Integration and Spring Batch make this possible
- Spring Boot simplifies it
- Spring XD/Spring Cloud Data Flow makes it even easier

History

- Project started by Dave Syer and Costin Leau in 2011
- First 1.0 GA release in February 2013
- Older versions:
  - 2.0.4 supports Hadoop v1 & v2 (Hadoop 1.2.1 - 2.6.0)
  - 2.1.0 Hadoop v2 only (Hadoop 2.2.0 - 2.6.0)
- Current version:
  - 2.2.0 Hadoop v2 (now Java 7+)
- Next version:
  - 2.3.0 Hadoop v2 (now HiveServer2, Spark)

A unified model for different Hadoop distros

- Spring for Apache Hadoop provides several “flavors” to match dependencies with Hadoop distributions from:
  - Apache Hadoop
  - Cloudera CDH
  - Hortonworks HDP
  - Pivotal HD

- See Wiki page for details:
  - ✓ https://github.com/spring-projects/spring-hadoop/wiki#supported-distributions
  - ✓ https://github.com/spring-projects/spring-hadoop/wiki#building-using-supported-distributions

Used in Spring XD and Spring Cloud Data Flow

- Spring XD
  - ✓ hdfs sink
  - ✓ hdfs-dataset sink
  - ✓ filepollhdfs
  - ✓ ftphdfs
  - ✓ hdfsjdbc
  - ✓ hdfsmongodb
  - ✓ jdbchdfs

- Spring Cloud Data Flow
  - ✓ hdfs sink

More modules coming soon!

Pro Tip:
Use separate JVMs for Spring XD 1.x modules that interleave or you might experience "java.io.IOException: Filesystem closed"

See XD-2505

What is new in 2.3?

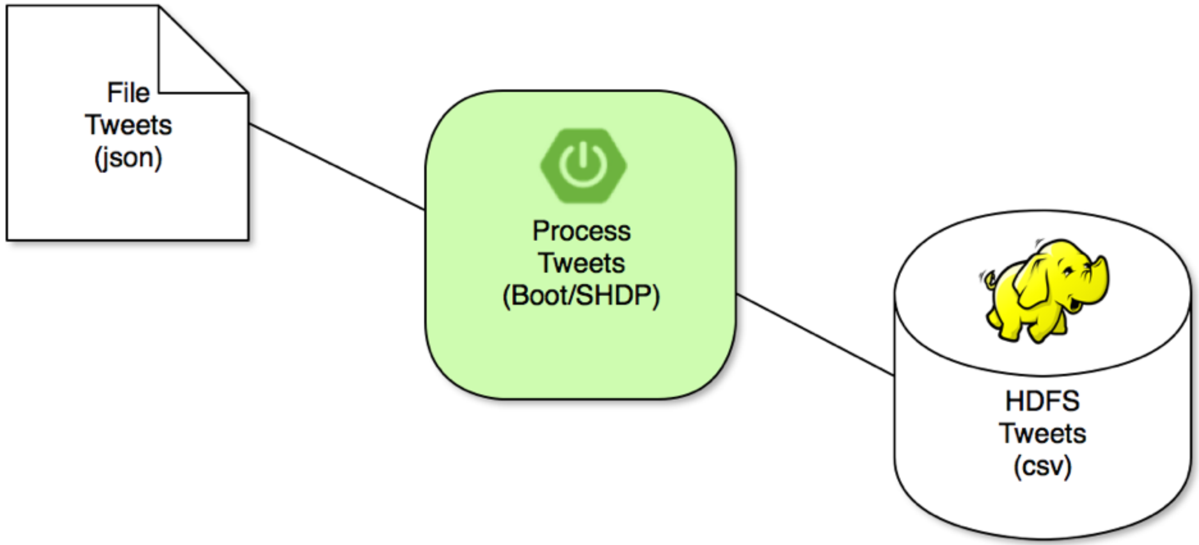
- Build and versions
  - Spring 4.2 and Spring Boot 1.3
  - Apache Hadoop 2.7.1 is now the default
  - Support for HDP 2.3 and Cloudera CDH 5.4
  - Support for Hive 1.x
- Features
  - Improved `@Configuration` support
  - HiveServer2 / JDBC replaces HiveServer1 / Thrift
    - see https://github.com/spring-projects/spring-hadoop-samples/commit/b1569e5f9f1fdfde9530e44bf0b32c0d1d3798d1
  - New Spark tasklet for executing Spark apps on YARN
  - New Sqoop2 tasklet for running Sqoop2 jobs

Data Ingestion

Many options for Data Ingestion

- Hadoop utilities: FileSystem Shell / Flume / Sqoop
- Spring XD
- Spring for Apache Hadoop's `FSShell`
- Spring Batch job
- Spring Boot app using a `DataStoreWriter` implementation

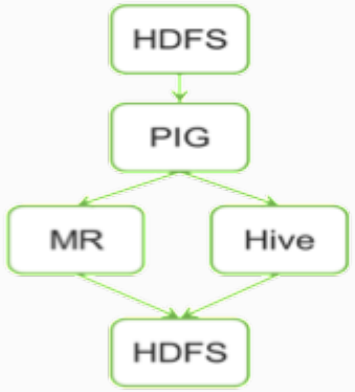
Demo #1 - Ingestion



Batch Processing

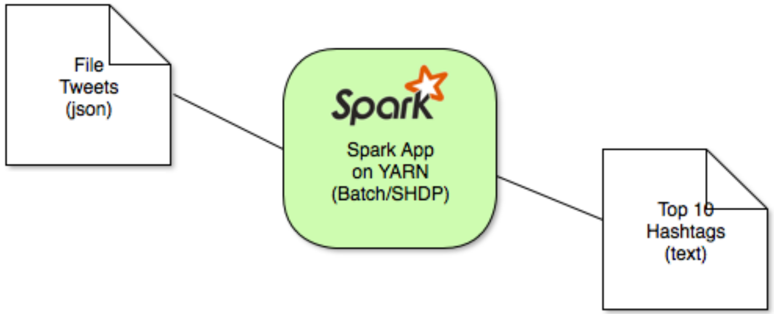
Spring Batch Highlights

- Spring Batch is a Framework for batch processing
  - Basis for JSR-352
- Well suited for multi-step processing
  - conditional control logic
  - parallell execution
  - restart failed steps
- Tracks job progress in a repository
- Supports many Hadoop based workloads in addition to non-Hadoop processing like File or JDBC based jobs
- Integrates with Spring Boot using `@EnableBatchProcessing`

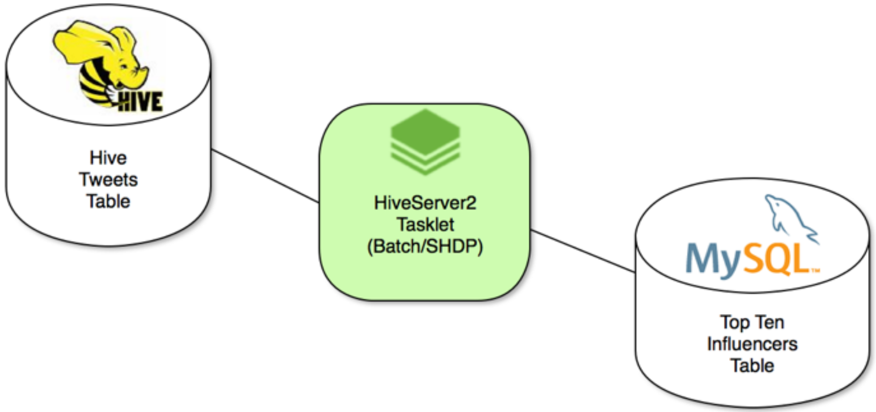

$$\begin{tikzcd} & \text{HDFS} \arrow[d] & \\ & \text{PIG} \arrow[dl] \arrow[dr] & \\ \text{MR} \arrow[dr] & & \text{Hive} \arrow[dl] \\ & \text{HDFS} & \end{tikzcd}$$

Batch Tasklets for Hadoop

Demo #2 - Batch Tasklets



SparkYarnTasklet



HiveServer2Tasklet + RDBMS Export

Cloud Native

Using Hadoop in the Cloud

- Amazon Elastic MapReduce
- Microsoft Azure HDInsight
- IBM BigInsights
- Hortonworks/SequenceIQ Cloudbreak
- Cloudera on AWS / Cloudera Live
- Your own Docker image
- Your own AWS installation

Common issues with Hadoop Cloud Clusters

- Hadoop has a cluster centric view
  - easier to run apps from inside the cluster
  - you should have core-site.xml, yarn-site.xml etc accessible
  - some insights into internal configs might be necessary
- Spring for Apache Hadoop tries to work around this
  - creating its own Hadoop Configuration
    - pulling from environment and config properties
- Cloud clusters usually configured for internal network
  - hard/impossible to reach from outside

Connecting to Hadoop in a cloud environment

- Network issues
  - are both NameNode and DataNodes visible from app
  - work arounds:
    - SOCKS proxy
    - docker --add-host borneo:192.168.55.9
- Configuration options
  - Spring profiles - spring.profiles.active=cloud
  - Env vars - spring_hadoop_fsUri=hdfs://borneo:8020
  - Spring Cloud Config Server
  - Some day maybe - spring-cloud-connectors for auto-reconfiguration

Use Hadoop with Cloud Foundry

- Deploy Hadoop separately
- Use a user-provided service:

```
cf create-user-provided-service hadoop -p \
  '{"fs":{"defaultFS":"hdfs://borneo:8020"},
    "yarn":{"resourcemanager":{"host":"borneo","port":"8050"}}}'
```

- Refer to the VCAP_SERVICES env var values in Boot config file:

```
---
spring:
  profiles: cloud
  hadoop:
    fsUri: ${vcap.services.hadoop.credentials.fs.defaultFS}
    resourceManagerHost: ${vcap.services.hadoop.credentials.yarn.resourcemanager.host}
    resourceManagerPort: ${vcap.services.hadoop.credentials.yarn.resourcemanager.port}
```

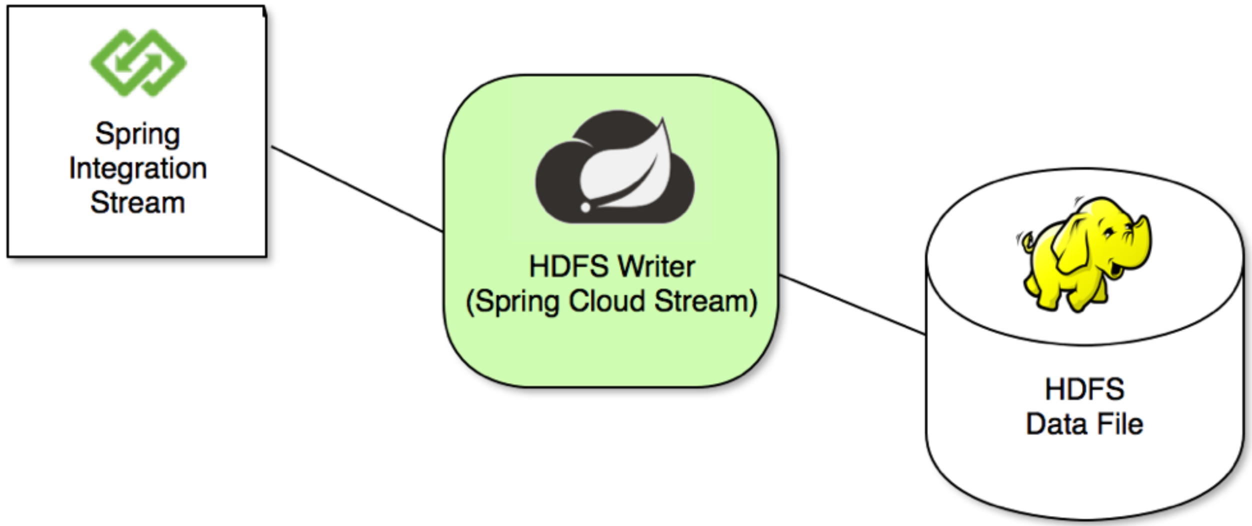
New Spring Cloud projects and Spring XD v2

- Spring Cloud Stream
  - Allows a user to develop and run data microservices using Spring Integration messaging and run them locally, in the cloud, or on Spring Cloud Data Flow
- Spring Cloud Stream Modules
  - pre-packaged modules for streaming workloads
- Spring Cloud Task Modules
  - pre-packaged modules for tasks and batch workloads
- Spring Cloud Data Flow (*Spring XD v2*)
  - Provides orchestration for data microservices, including spring-cloud-stream, spring-cloud-stream-modules and spring-cloud-task-modules.

Writing modules for Spring Cloud Stream

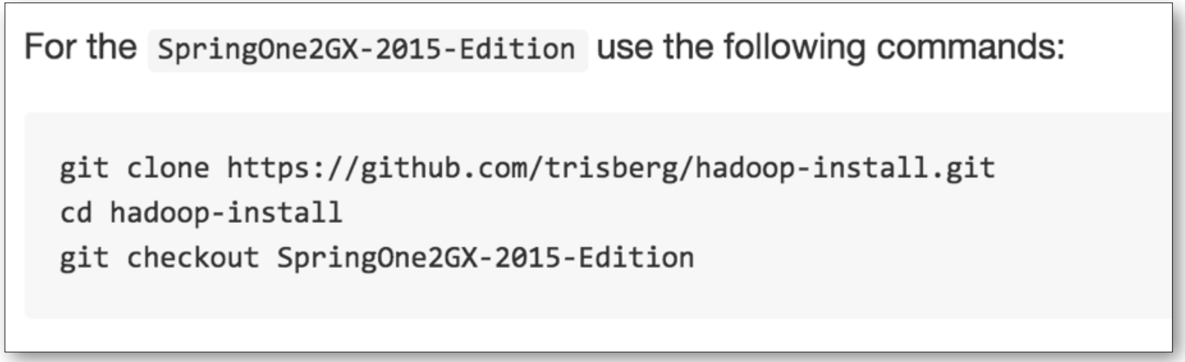
- Developing custom data modules
  - We can tap into the spring-cloud-stream infrastructure and either run separately or as part of Spring Cloud Data Flow.
- For a stream data module:
  - depend on spring-cloud-stream and a binder implementation (Redis/Rabbit/Kafka)
  - In addition to `@SpringBootApplication` use two new annotations:
    - `@EnableBinding` and `@ServiceActivator`

Demo #3 - Cloud-Native HDFS Writer App



Demo Environment - Hadoop

- Vagrant configuration for Hadoop:
  - Works on OS X, Linux and Windows
  - https://github.com/trisberg/hadoop-install
  - clone this repo and checkout a desired branch
    - (SpringOne2GX-2015-Edition)



For the `SpringOne2GX-2015-Edition` use the following commands:

```
git clone https://github.com/trisberg/hadoop-install.git
cd hadoop-install
git checkout SpringOne2GX-2015-Edition
```

Demo Environment - Windows

- To access Hadoop running on Linux from Windows client
  - You need a very minimal local Hadoop install
    - Download

      http://public-repo-1.hortonworks.com/hdp-win-alpha/winutils.exe
    - Place it in a bin directory under a Hadoop directory (C:\Hadoop\bin)
    - Then use: `java -D"hadoop.home.dir=C:\Hadoop" -jar …`
- Tested the following demos on Windows 8.1:
  - boot-ingest
  - batch-hive2
  - batch-spark

SPRINGONE2GX
WASHINGTON, DC

Learn More. Stay Connected.

@springcentral Spring.io/video

- Demo Source & Slides: https://github.com/trisberg/springone-2015
- Hadoop Install: https://github.com/trisberg/hadoop-install
- Spring for Apache Hadoop Project: http://projects.spring.io/spring-hadoop/
- Questions: http://stackoverflow.com/questions/tagged/spring-data-hadoop
- Twitter: @trisberg