Azkaban Documentation

Liang

Aug 03, 2018

Contents:

CHAPTER 1

What is Azkaban

Azkaban is a distributed Workflow Manager, implemented at LinkedIn to solve the problem of Hadoop job dependencies. We had jobs that needed to run in order, from ETL jobs to data analytics products.

See the *Getting Started* for more details.

CHAPTER 2

Features

- Distributed Multiple Executor
- MySQL Retry
- Friendly UI
- Conditional Workflow
- Data Triggers
- High Security
- Support plug-in extensions, from Web UI to job Execution
- Full Authorship management system

2.1 Getting Started

After version 3.0, we provide two modes: the stand alone "solo-server" mode and distributed multiple-executor mode. The following describes the differences between the two modes.

In solo server mode, the DB is embedded H2 and both web server and executor server run in the same process. This should be useful if one just wants to try things out. It can also be used on small scale use cases.

The multiple executor mode is for most serious production environment. Its DB should be backed by MySQL instances with master-slave set up. The web server and executor servers should ideally run in different hosts so that upgrading and maintenance shouldn't affect users. This multiple host setup brings in robust and scalable aspect to Azkaban.

- Set up the database
- Configure database to use multiple executors
- Download and install the Executor Server for each executor configured in database
- Install Azkaban Plugins
- Install the Web Server

Below are instructions on how to set Azkaban up.

2.1.1 Building from Source

Azkaban builds use Gradle (downloads automatically when run using gradlew which is the Gradle wrapper) and requires Java 8 or higher.

The following commands run on *nix platforms like Linux, OS X.

```
# Build Azkaban
./gradlew build

# Clean the build
./gradlew clean

# Build and install distributions
./gradlew installDist

# Run tests
./gradlew test

# Build without running tests
./gradlew build -x test
```

These are all standard Gradle commands. Please look at Gradle documentation for more info.

Gradle creates .tar.gz files inside project directories. eg. ./azkaban-solo-server/build/distributions/azkaban-solo-server-0.1.0-SNAPSHOT.tar.gz. Untar using tar -xvzf path/to/azkaban-*.tar.gz.

2.1.2 Getting started with the Solo Server

The solo server is a standalone instance of Azkaban and the simplest to get started with. The solo server has the following advantages.

- **Easy to install** - No MySQL instance is needed. It packages H2 as its main persistence storage.
- **Easy to start up** - Both web server and executor server run in the same process.
- **Full featured** - It packages all Azkaban features. You can use it in normal ways and install plugins for it.

Installing the Solo Server

Follow these steps to get started:

1. Clone the repo:

```
git clone https://github.com/azkaban/azkaban.git
```

2. Build Azkaban and create an installation package:

```
cd azkaban; ./gradlew build installDist
```

3. Start the solo server:

```
cd azkaban-solo-server/build/install/azkaban-solo-server; bin/azkaban-solo-start.sh
```

Azkaban solo server should be all set, by listening to `8081` port at default to accept incoming network request. So, open a web browser and check out `http://localhost:8081/`

4. Stop server:

```
bin/azkaban-solo-shutdown.sh
```

The solo-server installation should contain the following directories.

Folder	Description
bin	The scripts to start/stop Azkaban solo server
conf	The configuration files for Azkaban solo server
lib	The jar dependencies for Azkaban
extlib	Additional jars that are added to extlib will be added to Azkaban's classpath
plugins	the directory where plugins can be installed
web	The web (css, javascript, image) files for Azkaban web server

Inside the `conf` directory, there should be three files:

- `azkaban.private.properties` - Used by Azkaban to store secrets like Mysql password
- `azkaban.properties` - Used by Azkaban for runtime parameters
- `global.properties` - Global static properties that are passed as shared properties to every workflow and job.
- `azkaban-users.xml` - Used to add users and roles for authentication. This file is not used if the XmLUser-Manager is not set up to use it.

The `The azkaban.properties` file is the main configuration file.

Configuring HTTPS server (*Optional*)

Azkaban solo server by default doesn't use SSL. But you could set it up the same way in a stand alone web server. Here is how:

Azkaban web server supports SSL socket connectors, which means a keystore will have to be available. You can follow the steps to generate a valid jetty keystore provided at here. Once a keystore file has been created, Azkaban must be given its location and password. Within `azkaban.properties` or `azkaban.private.properties` (recommended), the following properties should be overridden.

```
jetty.keystore=keystore
jetty.password=password
jetty.keypassword=password
jetty.truststore=keystore
jetty.trustpassword=password
```

And configure ssl port in *azkaban.properties*:

```
jetty.ssl.port=8443
```

2.1.3 Getting started with the Multi Executor Server

Databasea setup

We suggest users to opt for **Mysql** as Azkaban database, because we build up a few Mysql connection enhancements to facilitate AZ set up, and strengthen service reliability:

- Install Mysql

  Installation of MySQL DB won't be covered by these instructions, but you can access the instructions on MySQL Documentation Site.

- Set up Mysql

  1. create database for Azkaban.:

     ```
     # Example database creation command, although the db name doesn't need to be
     ↪'azkaban'
     mysql> CREATE DATABASE azkaban;
     ```

  2. create a mysql user for Azkaban. For example,:

     ```
     # Example database creation command. The user name doesn't need to be 'azkaban
     ↪'
     mysql> CREATE USER 'username'@'%' IDENTIFIED BY 'password';
     # give the user INSERT, SELECT, UPDATE, DELETE permission on all tables in␣
     ↪the Azkaban db.
     mysql> GRANT SELECT,INSERT,UPDATE,DELETE ON azkaban.* to '<username>'@'%'␣
     ↪WITH GRANT OPTION;
     ```

  3. Mysql Packet Size may need to be re-configured. MySQL may have, by default, a ridiculously low allowable packet size. To increase it, you'll need to have the property max_allowed_packet set to a higher number, say 1024M. To configure this in linux, open /etc/my.cnf. Somewhere after mysqld, add the following:

     ```
     [mysqld]
     ...
     max_allowed_packet=1024M
     ```

     To restart MySQL, you can run:

     ```
     $ sudo /sbin/service mysqld restart
     ```

- Create the Azkaban Tables

  Run individual table creation scripts from latest table statements on the MySQL instance to create your tables.

  Alternatively, run create-all-sql-<version>.sql generated by build process. The location is the file is at `/Users/latang/LNKDRepos/azkaban/azkaban-db/build/distributions/azkaban-db-<version>`, after you build *azkaban-db* module by

  ```
  cd azkaban-db; ../gradlew build installDist
  ```

Installing Azkaban Executor Server

Azkaban Executor Server handles the actual execution of the workflow and jobs. You can build the latest version from the master branch. See here for instructions on *Building from Source*.

Extract the package (executor distribution tar.gz from build folder) into a directory after gradle build. There should be the following directories.

Folder	Description
bin	The scripts to start/stop Azkaban solo server
conf	The configuration files for Azkaban solo server
lib	The jar dependencies for Azkaban
extlib	Additional jars that are added to extlib will be added to Azkaban's classpath
plugins	the directory where plugins can be installed

For quick start, we may directly use the Installation directory *azkaban/azkaban-exec-server/build/install/azkaban-exec-server* generated by gradle. we only need to change mysql username and password inside `azkaban.properties`:

```
# Mysql Configs
mysql.user=<username>
mysql.password=<password>
```

Then run:

```
cd azkaban-solo-server/build/install/azkaban-exec-server
./bin/start-exec.sh
```

After that, remember to activate the executor by calling:

```
cd azkaban-exec-server/build/install/azkaban-exec-server
curl -G "localhost:$(<./executor.port)/executor?action=activate" && echo
```

Then, one executor is ready for use. Users can set up multiple executors by distributing and deploying multiple executor installation distributions.

Installing Azkaban Web Server

Azkaban Web Server handles project management, authentication, scheduling and trigger of executions. You can build the latest version from the master branch. See here for instructions on *Building from Source*.

Extract the package (executor distribution tar.gz from build folder) into a directory after gradle build. There should be the following directories.

Folder	Description
bin	The scripts to start/stop Azkaban solo server
conf	The configuration files for Azkaban solo server
lib	The jar dependencies for Azkaban
web	The web (css, javascript, image) files for Azkaban web server

For quick start, we may directly use the Installation directory *azkaban/azkaban-web-server/build/install/azkaban-web-server* generated by gradle. we only need to change mysql username and password inside `azkaban.properties`:

```
# Mysql Configs
mysql.user=<username>
mysql.password=<password>
```

Then run

```
cd azkaban-web-server/build/install/azkaban-web-server
./bin/start-web.sh
```

Then, a multi-executor Azkaban instance is ready for use. Open a web browser and check out `http://localhost:8081/` You are all set to login to Azkaban UI.

2.2 Configurations

Azkaban can be configured in many ways. The following describes the knobs and switches that can be set. For the most part, there is no need to deviate from the default values.

2.2.1 Azkaban Web Server Configurations

These are properties to configure the web server. They should be set in `azkaban.properties`.

General Properties

Parameter	Description	Default
azkaban.name	The name of the azkaban instance that will show up in the UI. Useful if you run more than one Azkaban instance.	Local
azkaban.label	A label to describe the Azkaban instance.	My Local Azkaban
azkaban.color	Hex value that allows you to set a style color for the Azkaban UI.	#FF3601
web.resource.dir	Sets the directory for the ui's css and javascript files.	web/
default.timezone	The timezone that will be displayed by Azkaban.	America/Los_Angeles
viewer.plugin.dir	Directory where viewer plugins are installed.	plugins/viewer
job.max.Xms	The maximum initial amount of memory each job can request. This validation is performed at project upload time	1GB
job.max.Xmx	The maximum amount of memory each job can request. This validation is performed at project upload time	2GB

Multiple Executor Mode Parameters

Parameter	Description	Default
azkaban.use.multiple.executors	Should azkaban run in multi-executor mode. Required for multiple executor mode.	false
azkaban.executorselector.filters	A common separated list of hard filters to be used while dispatching. To be choosen from StaticRemaining, FlowSize, MinimumFreeMemory and CpuStatus. Order of filter do not matter.	
azkaban.executorselector.comparator.{ComparatorName}	Integer weight to be used to rank available executors for a given flow. Currently, {ComparatorName} can be NumberOfAssignedFlowC omparator, Memory, LastDispatched and CpuUsage as ComparatorName. For example:- azkaban.executorselector.comparator.Memory =2	
azkaban.queueprocessing.enabled	Hhould queue processor be enabled from webserver initialization	true
azkaban.webserver.queue.size	Maximum flows that can be queued at webserver	100000
azkaban.activeexecutor.refresh.milisecinterval	Maximum time in milliseconds that can be processed without executor statistics refresh	50000
azkaban.activeexecutor.refresh.flowinterval	Maximum number of queued flows that can be processed without executor statistics refresh	5
azkaban.executorinfo.refresh.maxThreads	Maximum number of threads to refresh executor statistics	5

Jetty Parameters

Parameter	Description	Default
jetty.maxThreads	Max request threads	25
jetty.ssl.port	The ssl port	8443
jetty.keystore	The keystore file	
jetty.password	The jetty password	
jetty.keypassword	The keypassword	
jetty.truststore	The trust store	
jetty.trustpassword	The trust password	

Project Manager Settings

Parameter	Description	Default
project.temp.dir	The temporary directory used when uploading projects	temp
project.version.reten tion	The number of unused project versions retained before cleaning	3
creator.default.proxy	Auto add the creator of the projects as a proxy user to the project.	true
lockdown.create.proje cts	Prevents anyone except those with Admin roles to create new projects.	false
lockdown.upload.proje cts	Prevents anyone but admin users and users with permissions to upload projects.	false

MySQL Connection Parameter

Parameter	Description	Default
database.type	The database type. Currently, the only database supported is mysql.	mysql
mysql.port	The port to the mysql db	3306
mysql.host	The mysql host	local-host
mysql.database	The mysql database	
mysql.user	The mysql user	
mysql.password	The mysql password	
mysql.numconnections	The number of connections that Azkaban web client can open to the database	100

Executor Manager Properties

Parameter	Description	Default
executor.port	The port for the azkaban executor server	12321
executor.host	The host for azkaban executor server	localhost
execution.logs.retent ion.ms	Time in milliseconds that execution logs are retained	7257600000L (12 weeks)

Notification Email Properties

Parameter	Description	Default
mail.sender	The email address that azkaban uses to send emails.	
mail.host	The email server host machine.	
mail.user	The email server user name.	
mail.password	The email password user name.	

User Manager Properties

Parameter	Description	Default
user.manager.class	The user manager that is used to authenticate a user. The default is an XML user manager, but it can be overwritten to support other authentication methods, such as JDNI.	azkaban.user.XmlUserM anager
user.manager.xml.file	Xml file for the XmlUserManager	conf/azkaban-users.xm l

User Session Properties

Parameter	Description	Default
session.time.to.live	The session time to live in ms seconds	86400000
max.num.sessions	The maximum number of sessions before people are evicted.	10000

2.2.2 Azkaban Executor Server Configuration

Executor Server Properties

Parameter	Description	Default
executor.port	The port for azkaban executor server	12321
executor.global.properties	A path to the properties that will be the parent for all jobs.	none
azkaban.execution.dir	The folder for executing working directories	executions
azkaban.project.dir	The folder for storing temporary copies of project files used for executions	projects
executor.flow.threads	The number of simulateous flows that can be run. These threads are mostly idle.	30
job.log.chunk.size	For rolling job logs. The chuck size for each roll over	5MB
job.log.backup.index	The number of log chunks. The max size of each logs is then the index * chunksize	4
flow.num.job.threads	The number of concurrent running jobs in each flow. These threads are mostly idle.	10
job.max.Xms	The maximum initial amount of memory each job can request. If a job requests more than this, then Azkaban server will not launch this job	1GB
job.max.Xmx	The maximum amount of memory each job can request. If a job requests more than this, then Azkaban server will not launch this job	2GB
azkaban.server.flow.max.running.minutes	The maximum time in minutes a flow will be living inside azkaban after being executed. If a flow runs longer than this, it will be killed. If smaller or equal to 0, there's no restriction on running time.	-1

MySQL Connection Parameter

Parameter	Description	Default
database.type	The database type. Currently, the only database supported is mysql.	mysql
mysql.port	The port to the mysql db	3306
mysql.host	The mysql host	localhost
mysql.database	The mysql database	
mysql.user	The mysql user	
mysql.password	The mysql password	
mysql.numconnection s	The number of connections that Azkaban web client can open to the database	100

2.2.3 Plugin Configurations

Execute-As-User

With a new security enhancement in Azkaban 3.0, Azkaban jobs can now run as the submit user or the user.to.proxy of the flow by default. This ensures that Azkaban takes advantage of the Linux permission security mechanism, and operationally this simplifies resource monitoring and visibility. Set up this behavior by doing the following:-

Execute.as.user is set to true by default. In case needed, it can also be configured to false in azkaban-plugin's commonprivate.properties Configure azkaban.native.lib= to the place where you are going to put the compiled execute-as-user.c file (see below) Generate an executable on the Azkaban box for azkaban-common/src/main/c/execute-as-user.c. **it should be named execute-as-user** Below is a sample approach

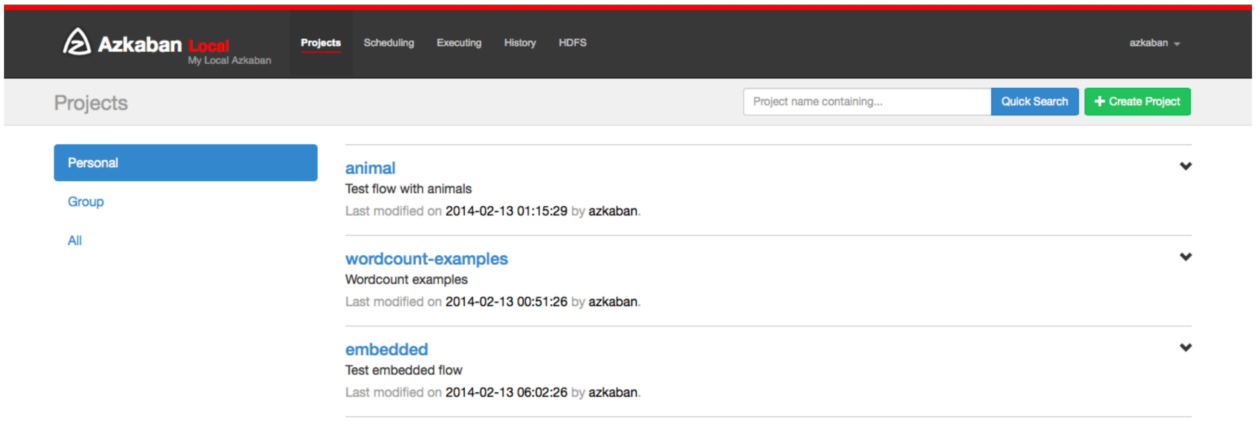
- `scp ./azkaban-common/src/main/c/execute-as-user.c` onto the Azkaban box
- run: `gcc execute-as-user.c -o execute-as-user`
- run: `chown root execute-as-user (you might need root privilege)`
- run: `chmod 6050 execute-as-user (you might need root privilege)`

2.3 Using Azkaban

This section covers how to use Azkaban Web UI to create, view and execution your flows.

2.3.1 Create Projects

After logging onto Azkaban, you will see the Projects page. This page will show you a list of all the projects that you have read permissions on. Projects where only group permissions as or those with a role with READ or ADMIN will not appear.



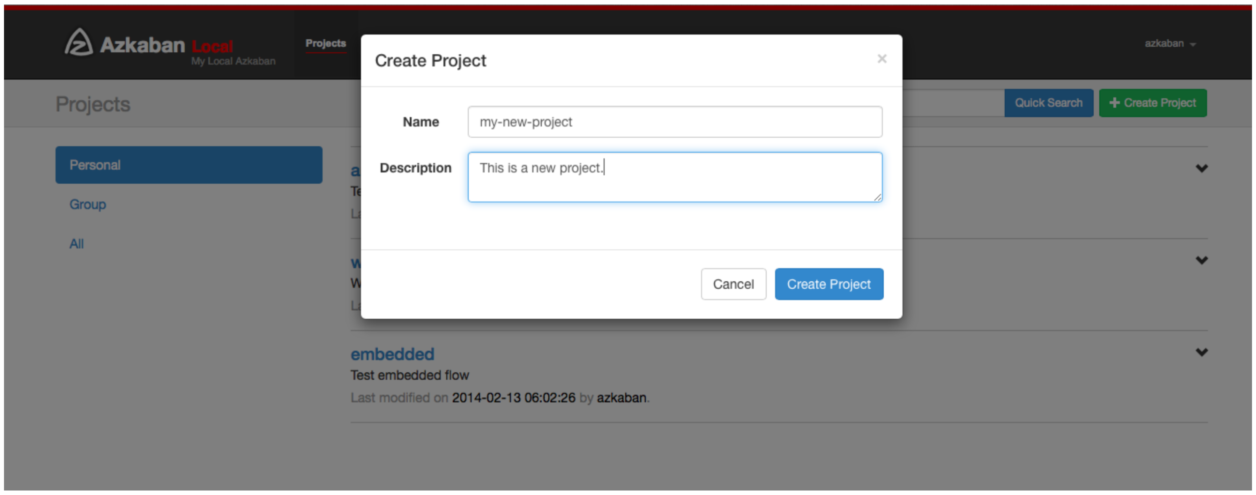
animal
Test flow with animals
Last modified on 2014-02-13 01:15:29 by azkaban.

wordcount-examples
Wordcount examples
Last modified on 2014-02-13 00:51:26 by azkaban.

embedded
Test embedded flow
Last modified on 2014-02-13 06:02:26 by azkaban.

If you are just starting out, the project page may be empty. However you can see all the existing projects by clicking on All Projects .

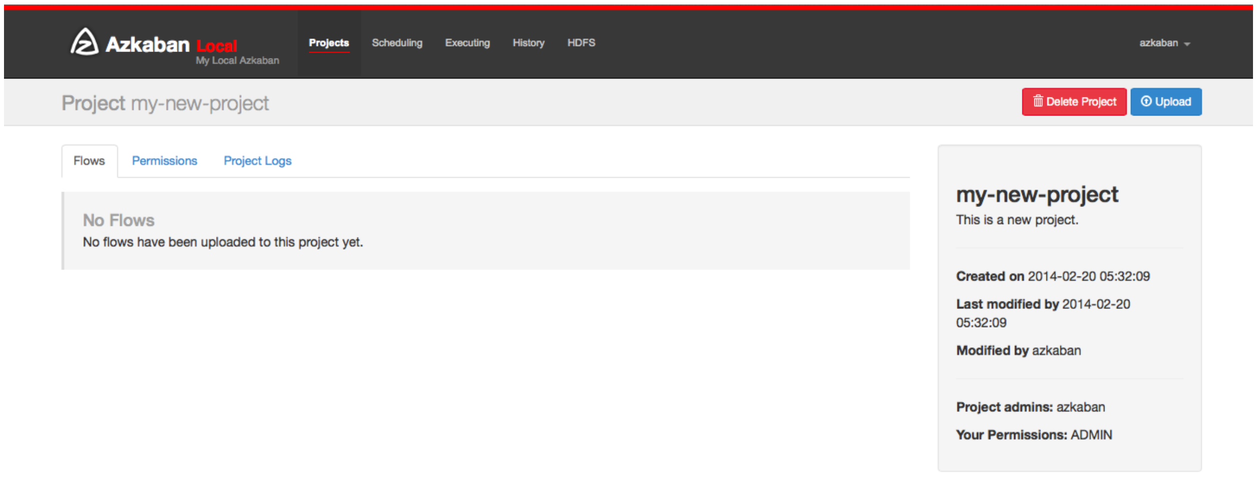
Clicking on **Create Projects** will pop open a dialog box. Enter in a unique project name and description of the project. The description can be changed in the future, but the project name cannot be. If you don't see this button, it is likely that the ability to create new projects has been locked down except for users with the proper permissions.



Name: my-new-project
Description: This is a new project.

embedded
Test embedded flow
Last modified on 2014-02-13 06:02:26 by azkaban.

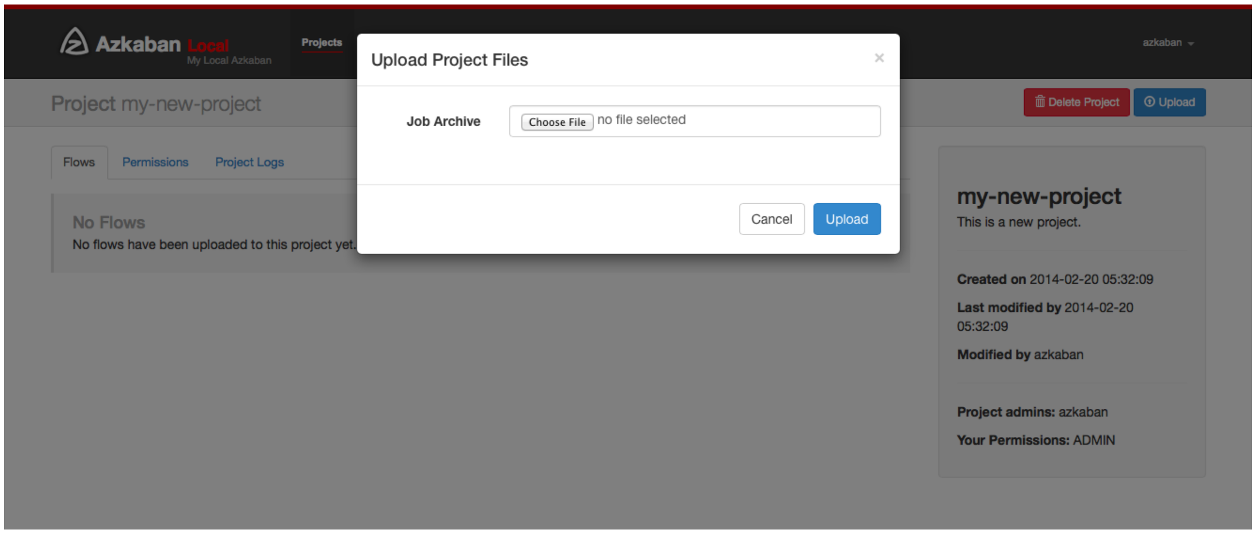
After creating your project, an empty project page will appear. You will automatically be given ADMIN status for this project. Add and remove permissions by clicking on the *Permissions Button*.



If you have the proper permissions (which you should if you created the project), you can delete the project, update the description, upload files and view the project logs from this page.

2.3.2 Upload Projects

Click on the **Upload** button. You will see the following dialog.



Select the archive file of your workflow files that you want to upload. Currently Azkaban only supports `*.zip` files. The zip should contain the `*.job` files and any files needed to run your jobs. Job names must be unique in a project.

Azkaban will validate the contents of the zip to make sure that dependencies are met and that there's no cyclical dependencies detected. If it finds any invalid flows, the upload will fail.

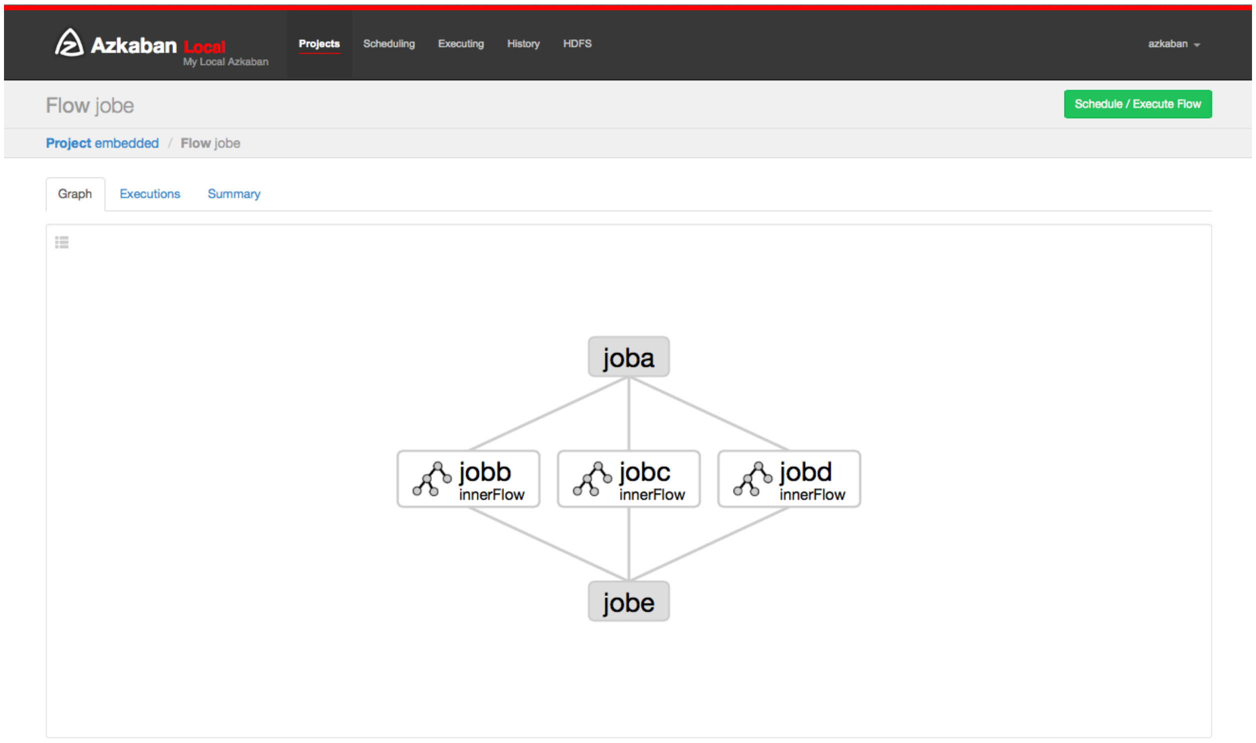
Uploads overwrite all files in the project. Any changes made to jobs will be wiped out after a new zip file is uploaded.

After a successful upload, you should see all of your flows listed on the screen.

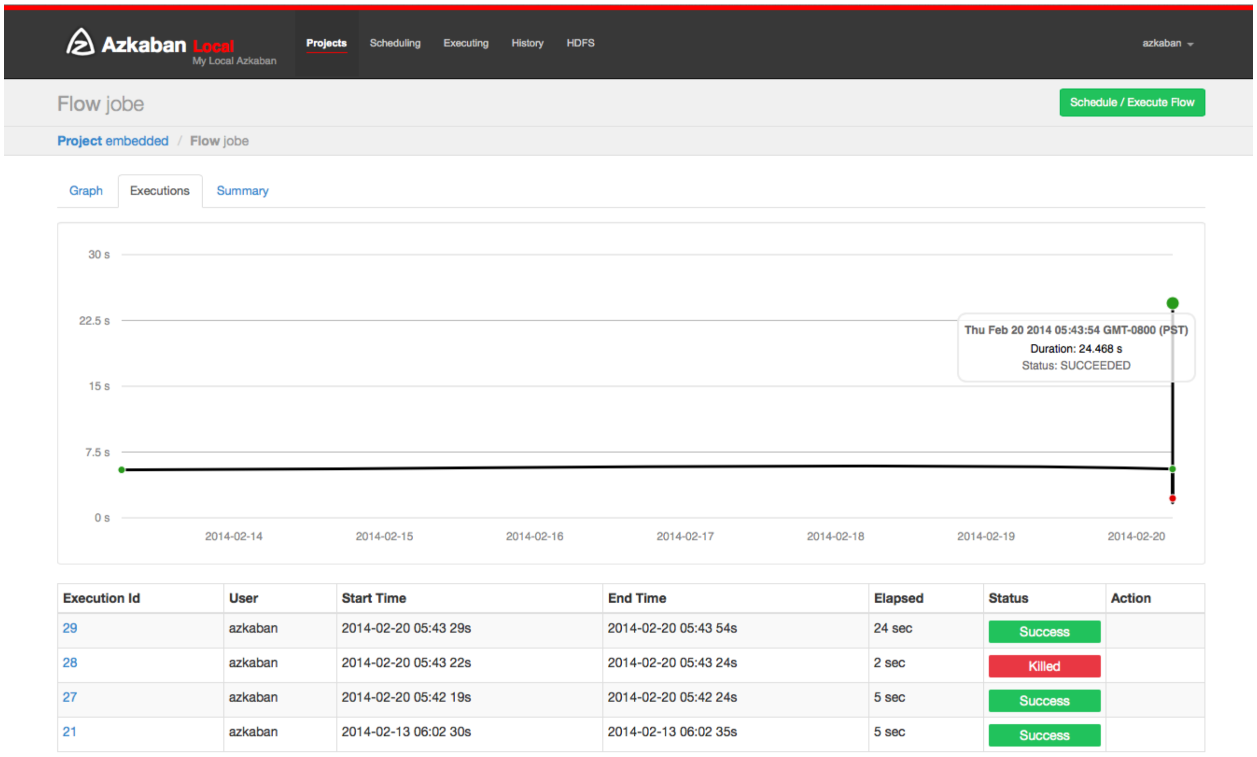
2.3.3 Flow View

By clicking on the flow link, you can go to the Flow View page. From here, you'll be shown a graph representation of the flow. The left panel contains a list of jobs in your flow.

Right clicking on either the jobs in the right panel or the nodes in the graph will allow you to open individual jobs. You are also able to Schedule and Execute Flows from this page.



Click on the Executions tab will show you all the previous executions of this flow.

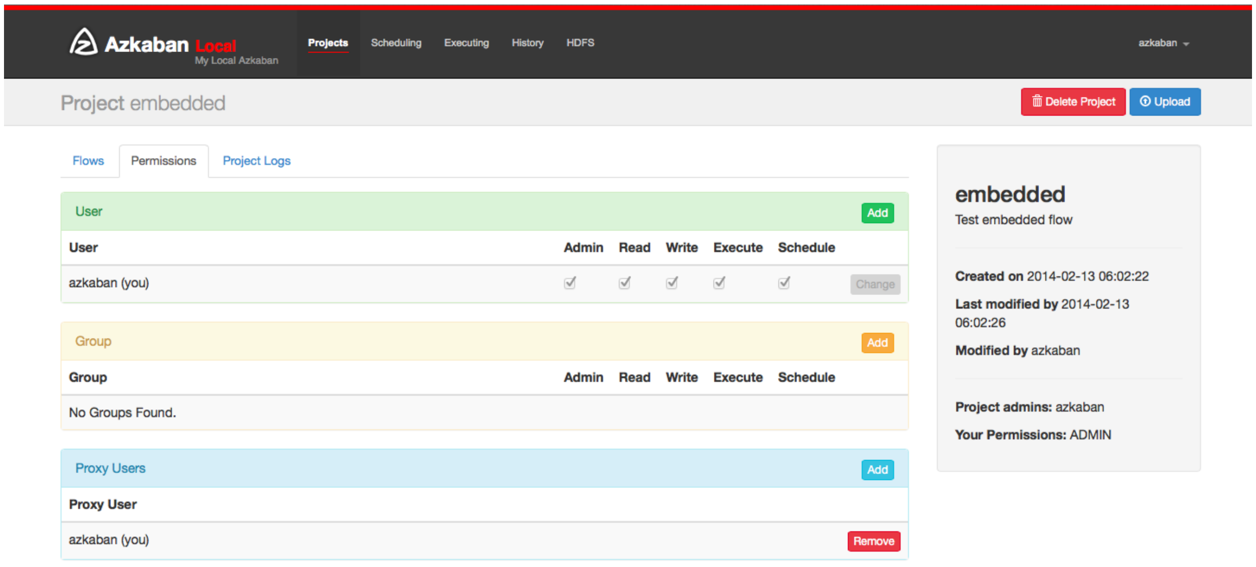


Execution Id	User	Start Time	End Time	Elapsed	Status	Action
29	azkaban	2014-02-20 05:43 29s	2014-02-20 05:43 54s	24 sec	Success	
28	azkaban	2014-02-20 05:43 22s	2014-02-20 05:43 24s	2 sec	Killed	
27	azkaban	2014-02-20 05:42 19s	2014-02-20 05:42 24s	5 sec	Success	
21	azkaban	2014-02-13 06:02 30s	2014-02-13 06:02 35s	5 sec	Success	

2.3.4 Project Permissions

When a project is created, the creator is automatically given an ADMIN status on the project. This allows the creator to view, upload, change jobs, run flows, delete and add user permissions to the project. An admin can remove other admins, but cannot remove themselves. This prevents projects from being admin-less except when admins are deleted by a user with an admin role.

The permission page is accessible from the project page. On the permissions page, admins can add other users, groups or proxy users to the project.



User	Admin	Read	Write	Execute	Schedule	
azkaban (you)	✓	✓	✓	✓	✓	Change

Group	Admin	Read	Write	Execute	Schedule
No Groups Found.					

Proxy User	
azkaban (you)	Remove

embedded
Test embedded flow

Created on 2014-02-13 06:02:22
Last modified by 2014-02-13 06:02:26
Modified by azkaban

Project admins: azkaban
Your Permissions: ADMIN

- Adding user permissions gives those users those specified permissions on the project. Remove user permissions by unchecking all of the permissions.
- Group permissions allow everyone in a particular group the specified permissions. Remove group permissions by unchecking all the group permissions.
- If proxy users are turned on, proxy users allows the project workflows to run as those users. This is useful for locking down which headless accounts jobs can proxy to. They are removed by clicking on the 'Remove' button once added.

Every user is validated through the UserManager to prevent invalid users from being added. Groups and Proxy users are also check to make sure they are valid and to see if the admin is allowed to add them to the project.

The following permissions can be set for users and groups:

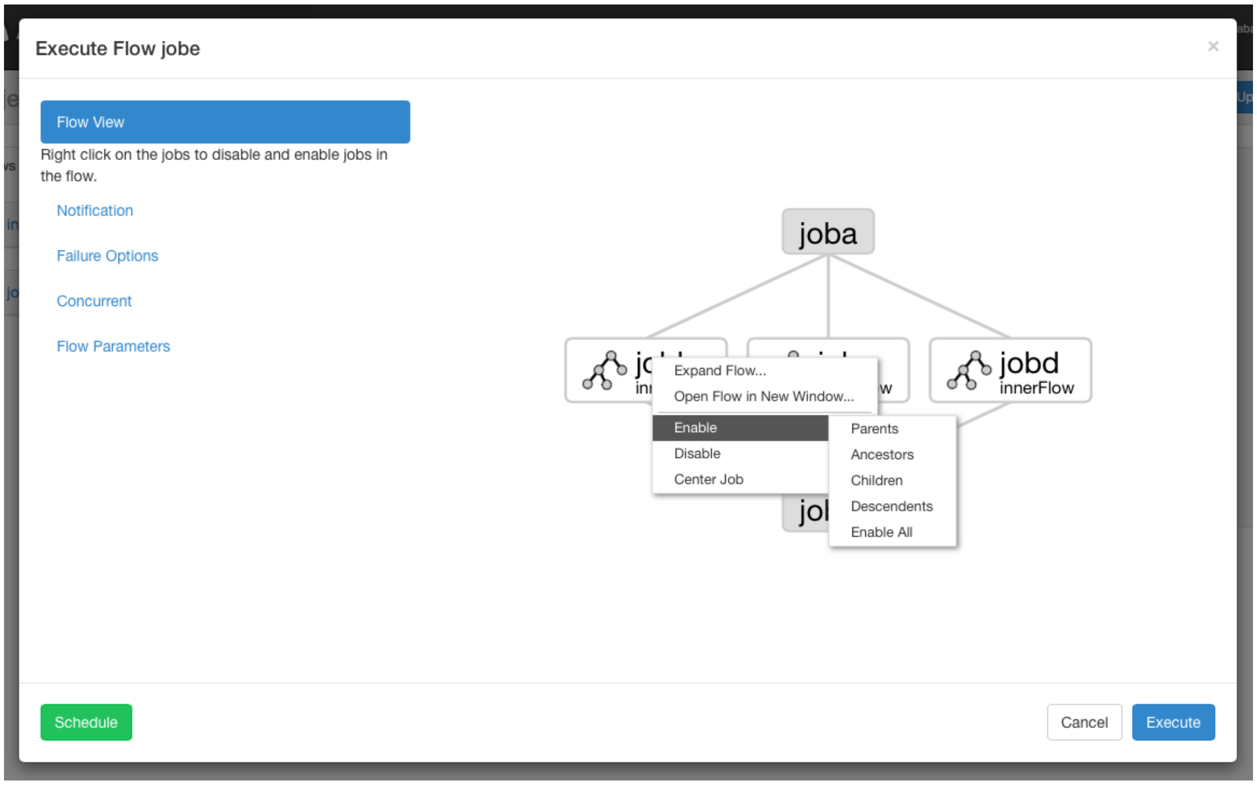
Permission	Description
ADMIN	Allows the user to do anything with this project, as well as add permissions and delete the project.
READ	The user can view the job, the flows, the execution logs.
WRITE	Project files can be uploaded, and the job files can be modified.
EXECUTE	The user is allowed to execute, pause, cancel jobs.
SCHEDULE	The user is allowed to add, modify and remove a flow from the schedule.

2.3.5 Executing Flow

From the *Flow View* or the Project Page, you can trigger a job to be executed. You will see an executing panel pop-up.

Executing Flow View

From the Flow View panel, you can right click on the graph and disable or enable jobs. Disabled jobs will be skipped during execution as if their dependencies have been met. Disabled jobs will appear translucent.



Notification Options

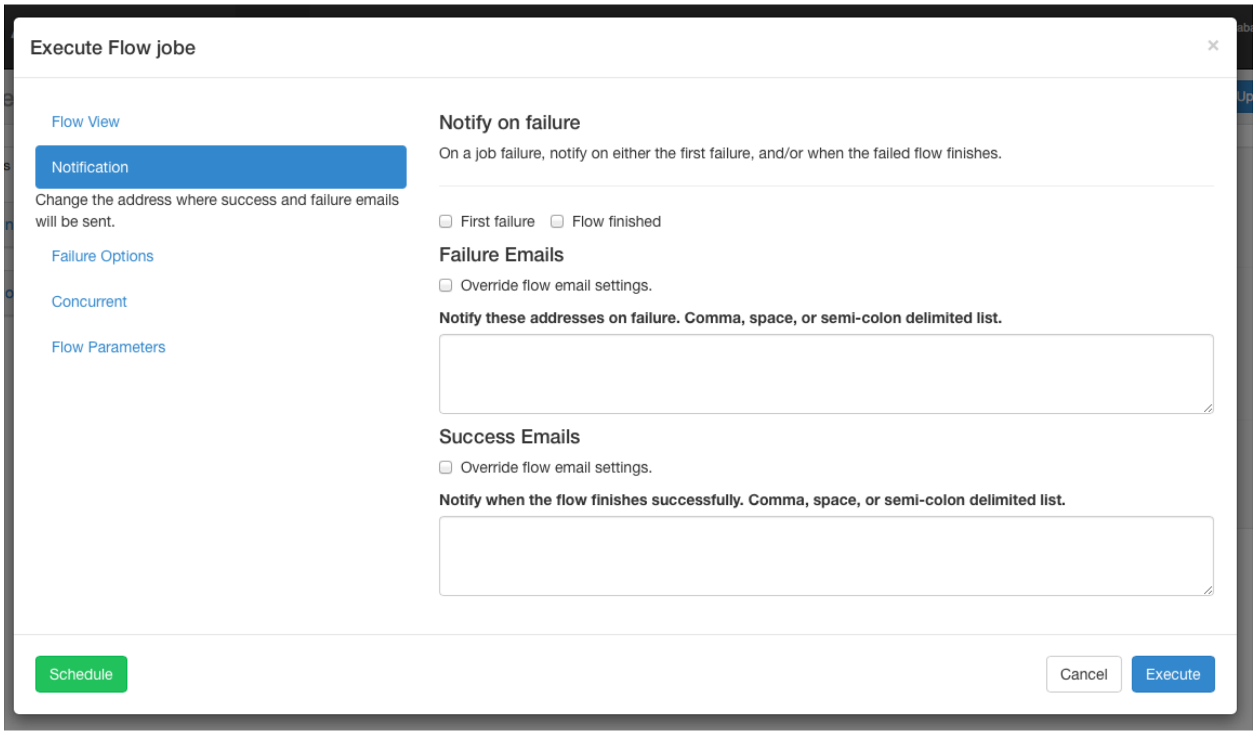
The notification options allow users to change the flow's success or failure notification behavior.

Notify on Failure

- First Failure - Send failure emails after the first failure is detected.
- Flow Finished - If the flow has a job that has failed, it will send failure emails after all jobs in the flow have finished.

Email overrides

Azkaban will use the default notification emails set in the final job in the flow. If overridden, a user can change the email addresses where failure or success emails are sent. The list can be delimited by commas, whitespace or a semi-colon.



Execute Flow jobe

Flow View

Notification

Change the address where success and failure emails will be sent.

Failure Options

Concurrent

Flow Parameters

Notify on failure

On a job failure, notify on either the first failure, and/or when the failed flow finishes.

☐ First failure ☐ Flow finished

Failure Emails

☐ Override flow email settings.

Notify these addresses on failure. Comma, space, or semi-colon delimited list.

Success Emails

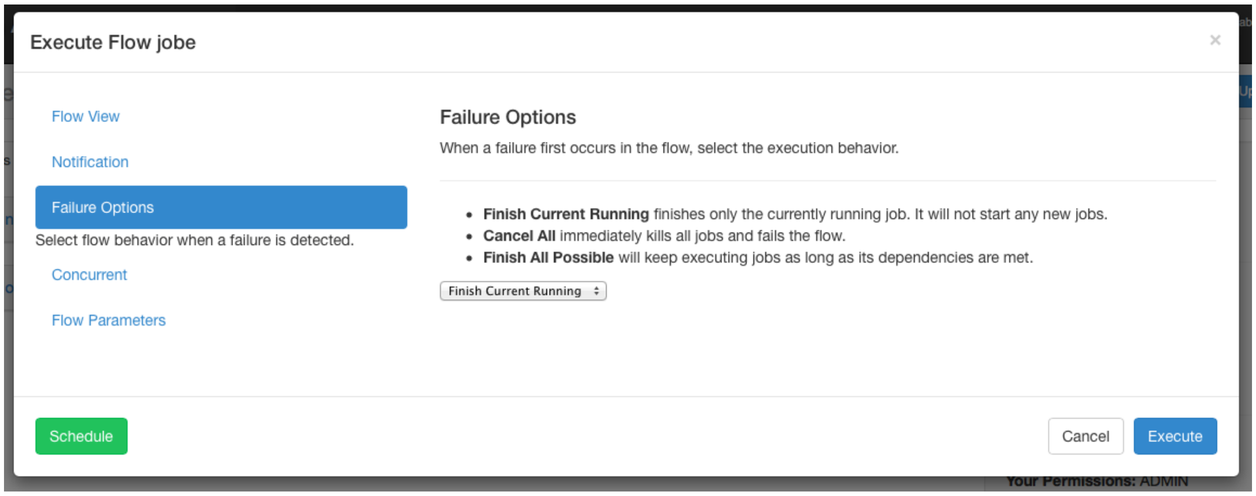
☐ Override flow email settings.

Notify when the flow finishes successfully. Comma, space, or semi-colon delimited list.

Failure Options

When a job in a flow fails, you are able to control how the rest of the flow will succeed.

- **Finish Current Running** will finish the jobs that are currently running, but it will not start new jobs. The flow will be put in the *FAILED FINISHING* state and be set to FAILED once everything completes.
- **Cancel All** will immediately kill all running jobs and set the state of the executing flow to FAILED.
- **Finish All Possible** will keep executing jobs in the flow as long as its dependencies are met. The flow will be put in the `FAILED FINISHING` state and be set to FAILED once everything completes.



Execute Flow jobe

Flow View

Notification

Failure Options

Select flow behavior when a failure is detected.

Concurrent

Flow Parameters

Failure Options

When a failure first occurs in the flow, select the execution behavior.

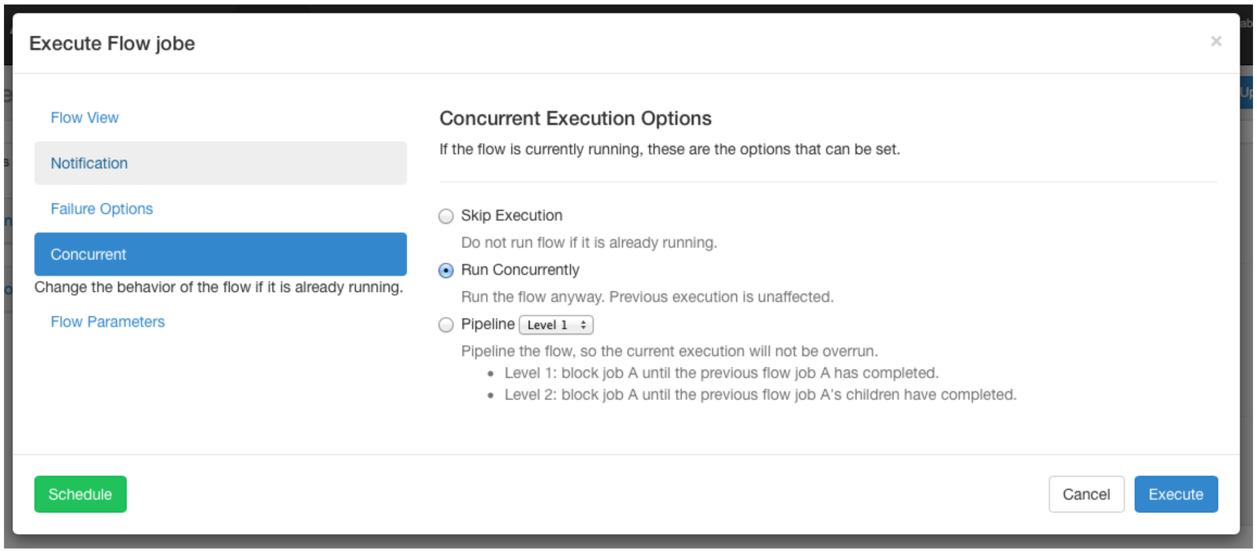
- **Finish Current Running** finishes only the currently running job. It will not start any new jobs.
- **Cancel All** immediately kills all jobs and fails the flow.
- **Finish All Possible** will keep executing jobs as long as its dependencies are met.

Finish Current Running

Concurrent Options

If the flow execution is invoked while the flow is concurrently executing, several options can be set.

- **Skip Execution** option will not run the flow if its already running.
- **Run Concurrently** option will run the flow regardless of if its running. Executions are given different working directories.
- **Pipeline** runs the the flow in a manner that the new execution will not overrun the concurrent execution.
  - Level 1: blocks executing **job A** until the the previous flow's **job A** has completed.
  - Level 2: blocks executing **job A** until the the children of the previous flow's **job A** has completed. This is useful if you need to run your flows a few steps behind an already executin flow.



Change the behavior of the flow if it is already running.

Concurrent Execution Options

If the flow is currently running, these are the options that can be set.

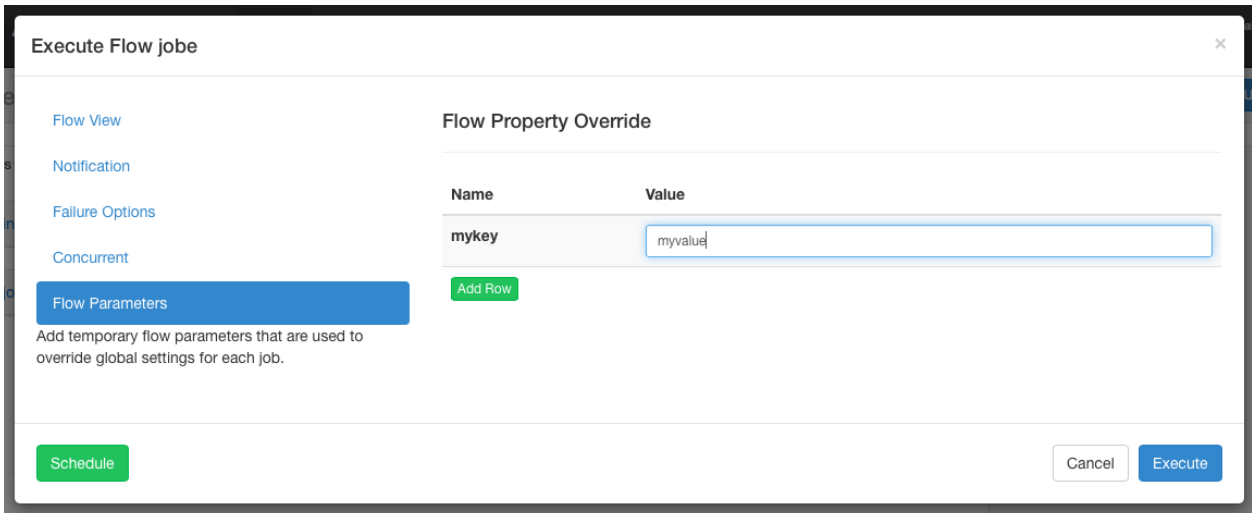
Skip Execution
Do not run flow if it is already running.

[✓ Run Concurrently]
Run the flow anyway. Previous execution is unaffected.

Pipeline Level 1
Pipeline the flow, so the current execution will not be overrun.
- Level 1: block job A until the previous flow job A has completed.
- Level 2: block job A until the previous flow job A's children have completed.

Flow Parameters

Allows users to override flow parameters. The flow parameters override the global properties for a job, but not the properties of the job itself.



Add temporary flow parameters that are used to override global settings for each job.

Flow Property Override

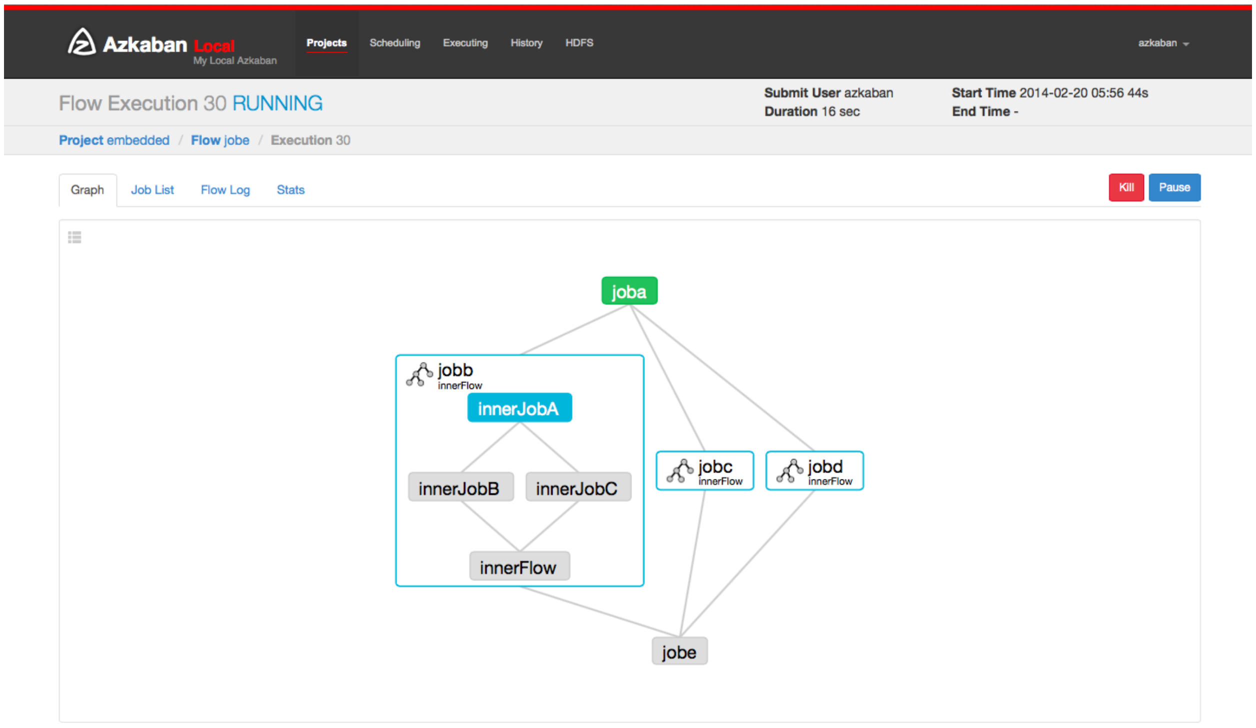
Name	Value
mykey	myvalue

2.3.6 Executions

Flow Execution Page

After *executing a flow* you will be presented the Executing Flow page. Alternatively, you can access these flows from the Flow View page under the Executions tab, the History page, or the Executing page.

This page is similar to the Flow View page, except it shows status of running jobs.

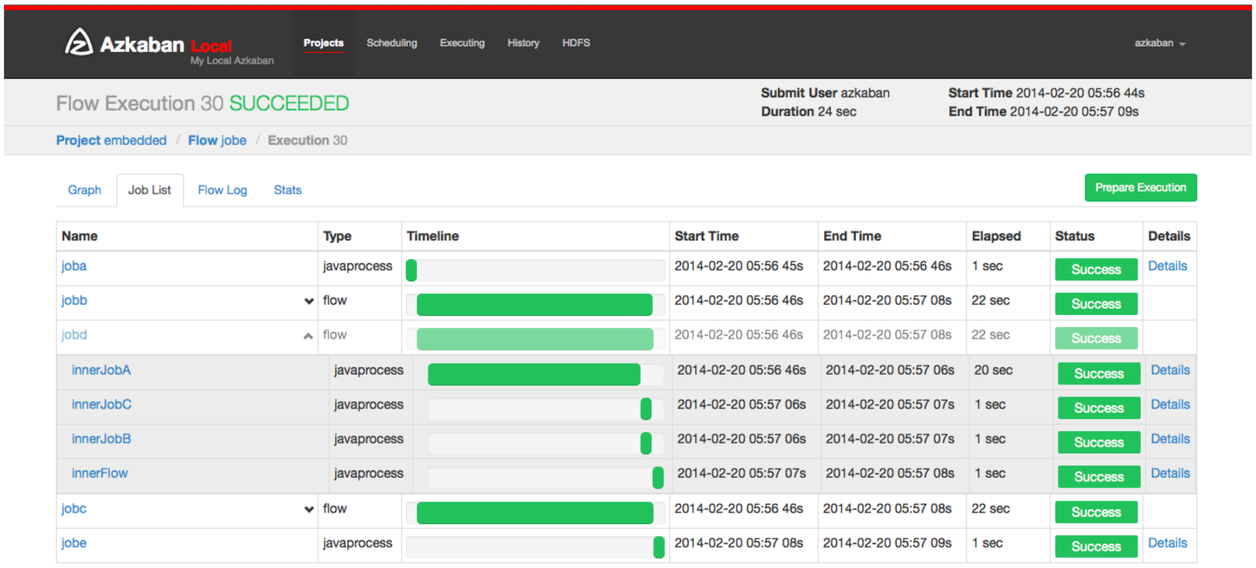


Flow Execution 30 RUNNING

Submit User azkaban
Duration 16 sec
Start Time 2014-02-20 05:56 44s
End Time -

Project embedded / **Flow** jobe / **Execution** 30

Selecting the Job List will give a timeline of job executions. You can access the jobs and job logs directly from this list.



Flow Execution 30 SUCCEEDED

Submit User azkaban
Duration 24 sec
Start Time 2014-02-20 05:56 44s
End Time 2014-02-20 05:57 09s

Project embedded / **Flow** jobe / **Execution** 30

Name	Type	Timeline	Start Time	End Time	Elapsed	Status	Details
joba	javaprocess		2014-02-20 05:56 45s	2014-02-20 05:56 46s	1 sec	Success	Details
jobb	flow		2014-02-20 05:56 46s	2014-02-20 05:57 08s	22 sec	Success	
jobd	flow		2014-02-20 05:56 46s	2014-02-20 05:57 08s	22 sec	Success	
innerJobA	javaprocess		2014-02-20 05:56 46s	2014-02-20 05:57 06s	20 sec	Success	Details
innerJobC	javaprocess		2014-02-20 05:57 06s	2014-02-20 05:57 07s	1 sec	Success	Details
innerJobB	javaprocess		2014-02-20 05:57 06s	2014-02-20 05:57 07s	1 sec	Success	Details
innerFlow	javaprocess		2014-02-20 05:57 07s	2014-02-20 05:57 08s	1 sec	Success	Details
jobc	flow		2014-02-20 05:56 46s	2014-02-20 05:57 08s	22 sec	Success	
jobe	javaprocess		2014-02-20 05:57 08s	2014-02-20 05:57 09s	1 sec	Success	Details

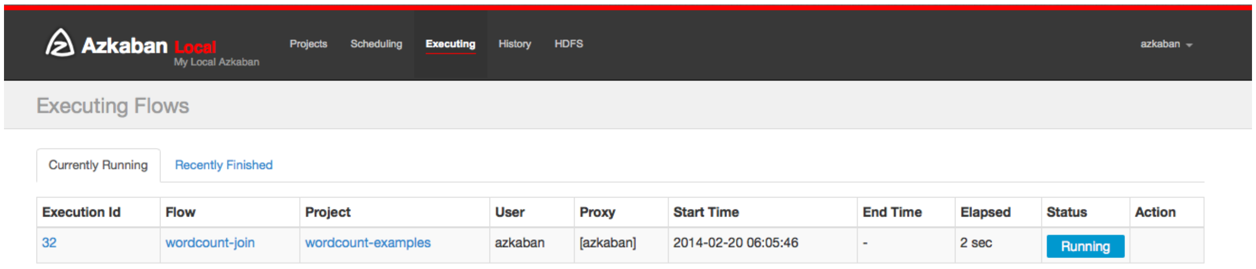
This page will auto update as long as the execution is not finished.

Some options that you are able to do on execution flows include the following:

- Cancel - kills all running jobs and fails the flow immediately. The flow state will be KILLED.
- Pause - prevents new jobs from running. Currently running jobs proceed as usual.
- Resume - resume a paused execution.
- Retry Failed - only available when the flow is in a FAILED FINISHING state. Retry will restart all FAILED jobs while the flow is still active. Attempts will appear in the Jobs List page.
- Prepare Execution - only available on a finished flow, regardless of success or failures. This will auto disable successfully completed jobs.

Executing Page

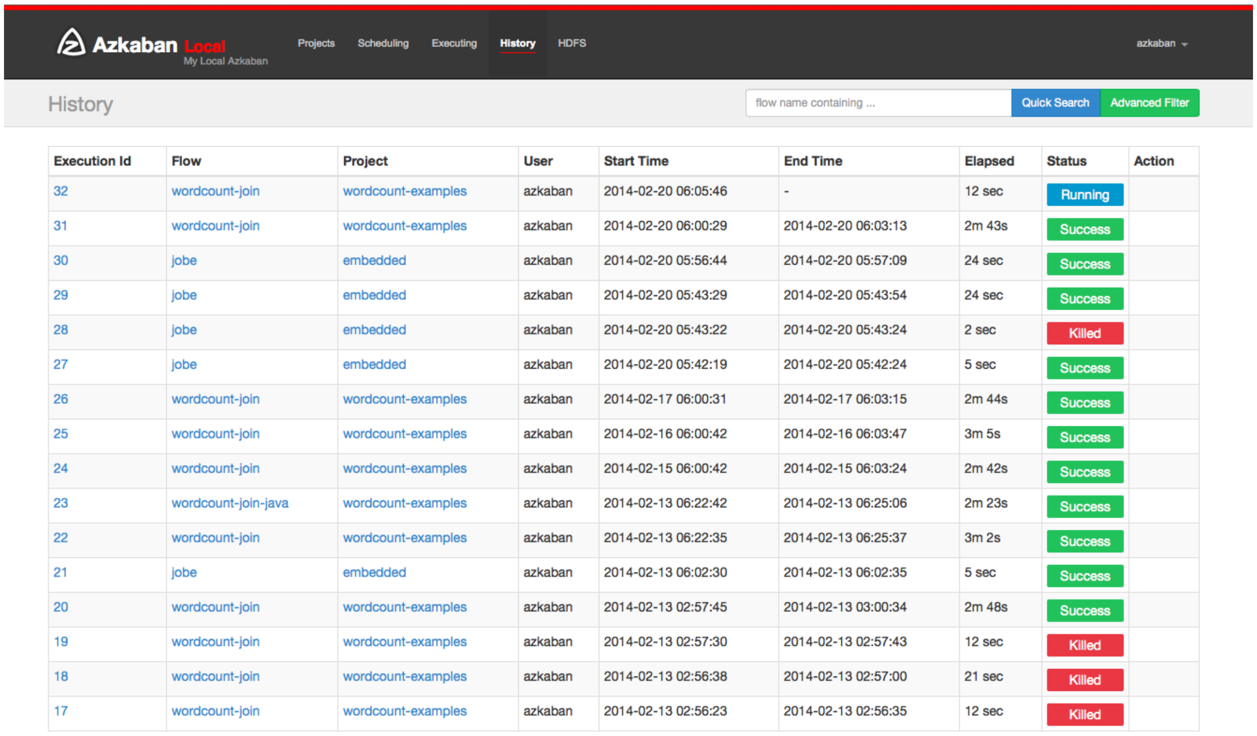
Clicking on the Executing Tab in the header will show the Execution page. This page will show currently running executions as well as recently finished flows.



Execution Id	Flow	Project	User	Proxy	Start Time	End Time	Elapsed	Status	Action
32	wordcount-join	wordcount-examples	azkaban	[azkaban]	2014-02-20 06:05:46	-	2 sec	Running	

History Page

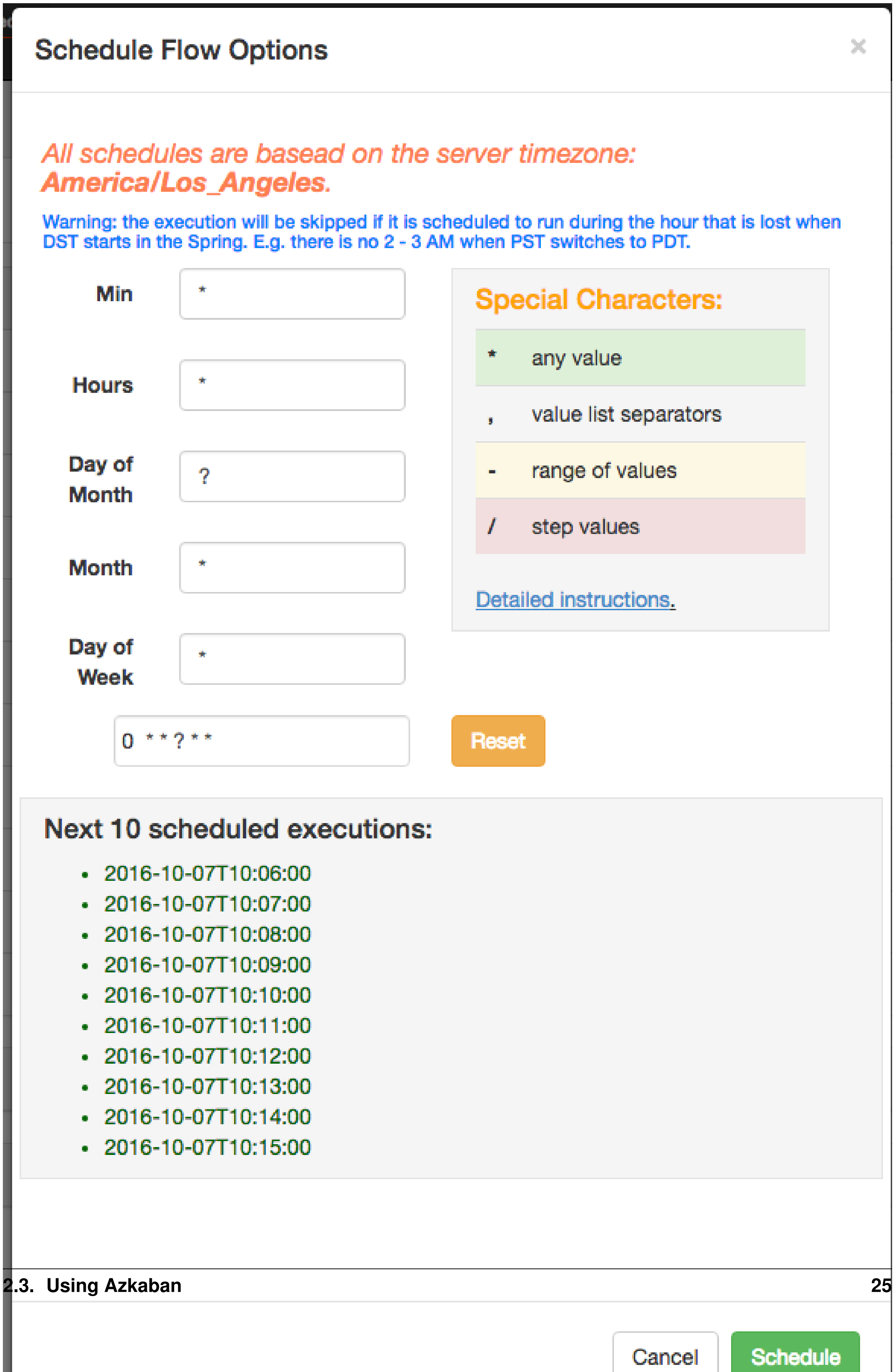
Currently executing flows as well as completed executions will appear in the History page. Searching options are provided to find the execution you're looking for. Alternatively, you can view previous executions for a flow on the Flow View execution tab.



Execution Id	Flow	Project	User	Start Time	End Time	Elapsed	Status	Action
32	wordcount-join	wordcount-examples	azkaban	2014-02-20 06:05:46	-	12 sec	Running	
31	wordcount-join	wordcount-examples	azkaban	2014-02-20 06:00:29	2014-02-20 06:03:13	2m 43s	Success	
30	jobe	embedded	azkaban	2014-02-20 05:56:44	2014-02-20 05:57:09	24 sec	Success	
29	jobe	embedded	azkaban	2014-02-20 05:43:29	2014-02-20 05:43:54	24 sec	Success	
28	jobe	embedded	azkaban	2014-02-20 05:43:22	2014-02-20 05:43:24	2 sec	Killed	
27	jobe	embedded	azkaban	2014-02-20 05:42:19	2014-02-20 05:42:24	5 sec	Success	
26	wordcount-join	wordcount-examples	azkaban	2014-02-17 06:00:31	2014-02-17 06:03:15	2m 44s	Success	
25	wordcount-join	wordcount-examples	azkaban	2014-02-16 06:00:42	2014-02-16 06:03:47	3m 5s	Success	
24	wordcount-join	wordcount-examples	azkaban	2014-02-15 06:00:42	2014-02-15 06:03:24	2m 42s	Success	
23	wordcount-join-java	wordcount-examples	azkaban	2014-02-13 06:22:42	2014-02-13 06:25:06	2m 23s	Success	
22	wordcount-join	wordcount-examples	azkaban	2014-02-13 06:22:35	2014-02-13 06:25:37	3m 2s	Success	
21	jobe	embedded	azkaban	2014-02-13 06:02:30	2014-02-13 06:02:35	5 sec	Success	
20	wordcount-join	wordcount-examples	azkaban	2014-02-13 02:57:45	2014-02-13 03:00:34	2m 48s	Success	
19	wordcount-join	wordcount-examples	azkaban	2014-02-13 02:57:30	2014-02-13 02:57:43	12 sec	Killed	
18	wordcount-join	wordcount-examples	azkaban	2014-02-13 02:56:38	2014-02-13 02:57:00	21 sec	Killed	
17	wordcount-join	wordcount-examples	azkaban	2014-02-13 02:56:23	2014-02-13 02:56:35	12 sec	Killed	

2.3.7 Schedule Flow

From the same panel that is used to *execute flow*, flows can be scheduled by clicking on the *Schedule* button.



Schedule Flow Options

All schedules are basead on the server timezone: ***America/Los_Angeles.***

Warning: the execution will be skipped if it is scheduled to run during the hour that is lost when DST starts in the Spring. E.g. there is no 2 - 3 AM when PST switches to PDT.

Min	*
Hours	*
Day of Month	?
Month	*
Day of Week	*

Special Characters:

*	any value
,	value list separators
-	range of values
/	step values

Detailed instructions.

0 * * ? * *

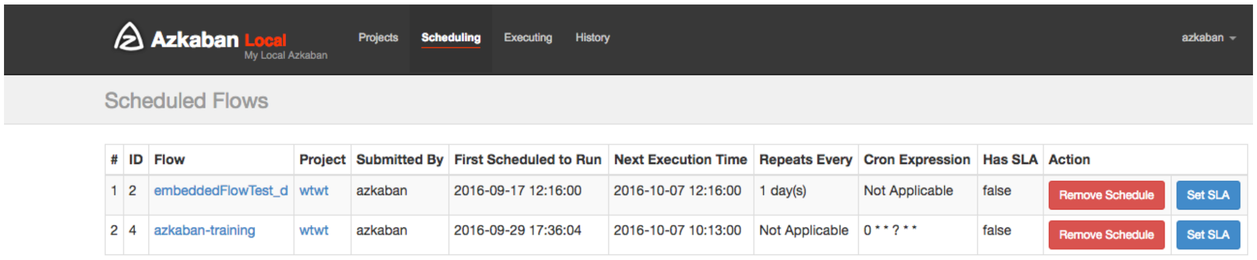
Next 10 scheduled executions:

- 2016-10-07T10:06:00
- 2016-10-07T10:07:00
- 2016-10-07T10:08:00
- 2016-10-07T10:09:00
- 2016-10-07T10:10:00
- 2016-10-07T10:11:00
- 2016-10-07T10:12:00
- 2016-10-07T10:13:00
- 2016-10-07T10:14:00
- 2016-10-07T10:15:00

Any flow options set will be preserved for the scheduled flow. For instance, if jobs are disabled, then the scheduled flow's jobs will also be disabled.

With new flexible scheduling feature in Azkaban 3.3, User are able to define a cron job following Quartz syntax. One important change different from Quartz or cron is that Azkaban functions at the minute granularity at most. Therefore, second field in UI is labeled as a static "0". The Flexible Schedule Wiki explains the details how to use.

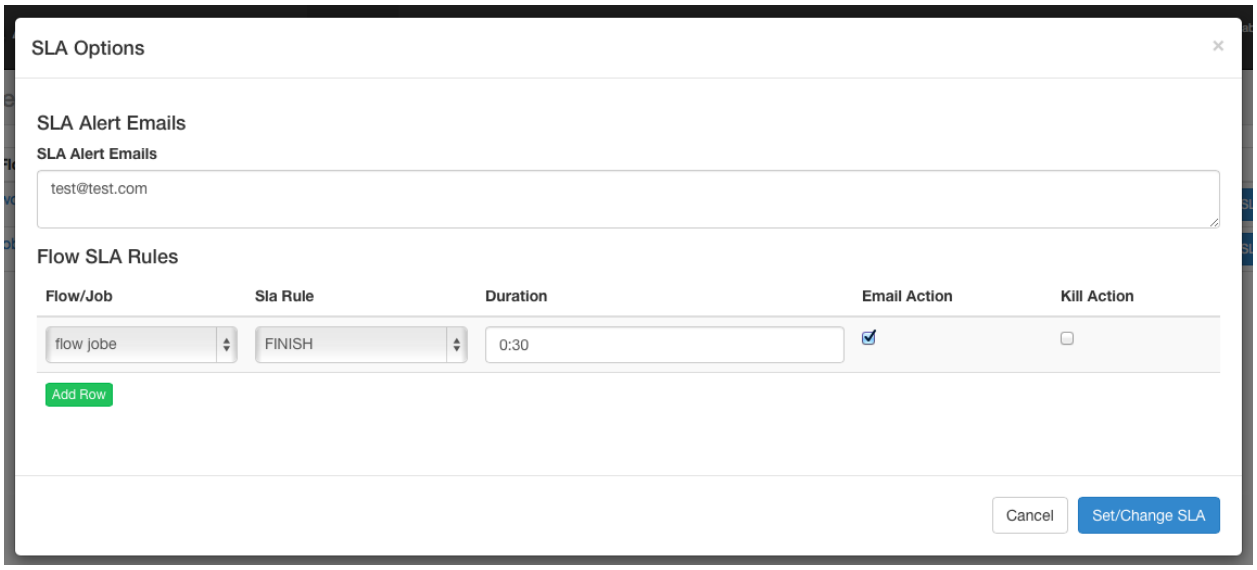
After scheduling, it should appear on the schedule page, where you can remove the scheduled job or set the SLA options.



#	ID	Flow	Project	Submitted By	First Scheduled to Run	Next Execution Time	Repeats Every	Cron Expression	Has SLA	Action	
1	2	embeddedFlowTest_d	wtwt	azkaban	2016-09-17 12:16:00	2016-10-07 12:16:00	1 day(s)	Not Applicable	false	Remove Schedule	Set SLA
2	4	azkaban-training	wtwt	azkaban	2016-09-29 17:36:04	2016-10-07 10:13:00	Not Applicable	0 * * ? * *	false	Remove Schedule	Set SLA

SLA

To add SLA notification or pre-emption, click on the SLA button. From here you can set the SLA alert emails. Rules can be added and applied to individual jobs or the flow itself. If duration threshold is exceeded, then an alert email can be set or the flow or job can be auto killed. If a job is killed due to missing the SLA, it will be retried based on the retry configuration of that job.



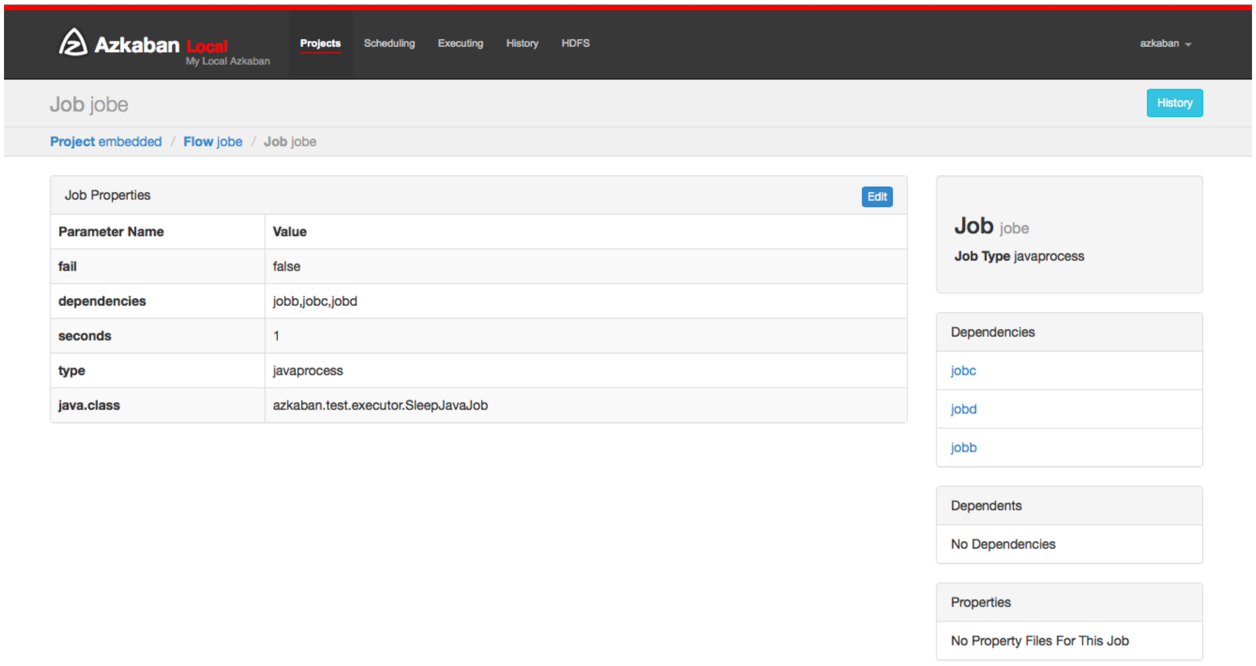
SLA Alert Emails

test@test.com

Flow/Job	Sla Rule	Duration	Email Action	Kill Action
flow jobe	FINISH	0:30	✓	☐

2.3.8 Job Page

Jobs make up individual tasks of a flow. To get to the jobs page, you can right click on a job in the Flow View, the Executing Flow view or the Project Page.



Job Properties

Parameter Name	Value
fail	false
dependencies	jobb,jobc,jobd
seconds	1
type	javaprocess
java.class	azkaban.test.executor.SleepJavaJob

Job jobe
Job Type javaprocess

Dependencies
jobc
jobd
jobb

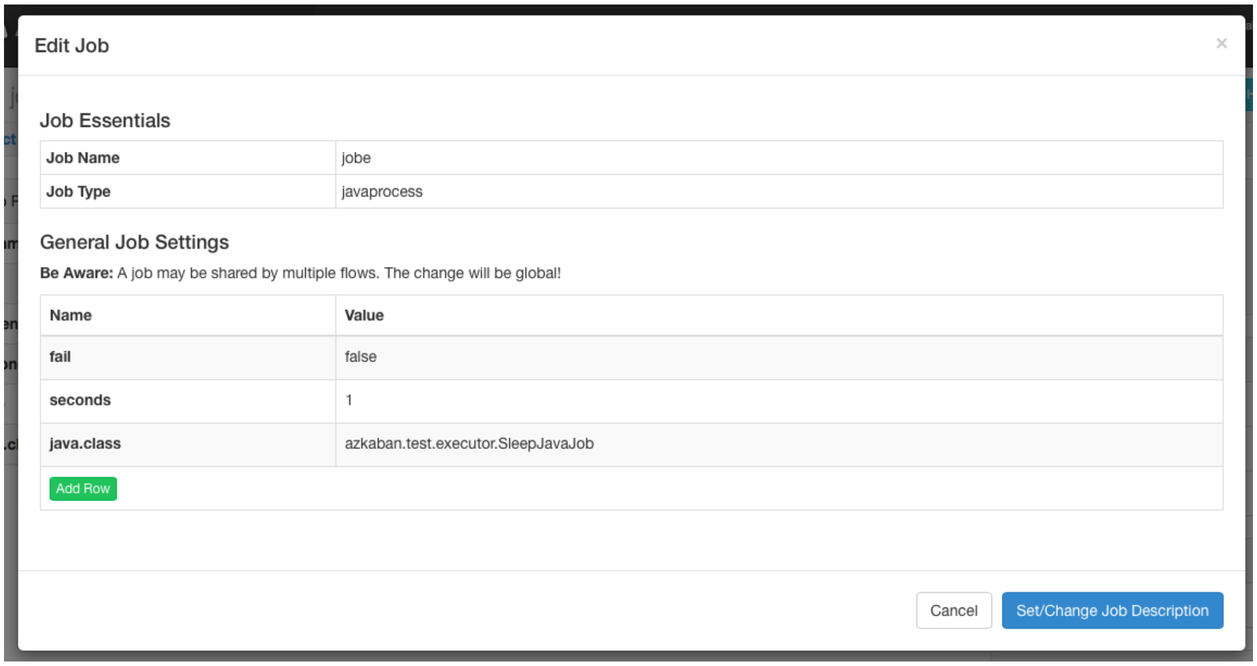
Dependents
No Dependencies

Properties
No Property Files For This Job

From this page you can see the dependencies and dependents for a job as well as the global properties that the job will use.

Job Edit

Clicking on Job Edit will allow you to edit all the job properties except for certain reserved parameters, such as `type`, and `dependencies`. The changes to the parameters will affect an executing flow only if the job hasn't started to run yet. These overwrites of job properties will be overwritten by the next project upload.



Job Essentials

Job Name	jobe
Job Type	javaprocess

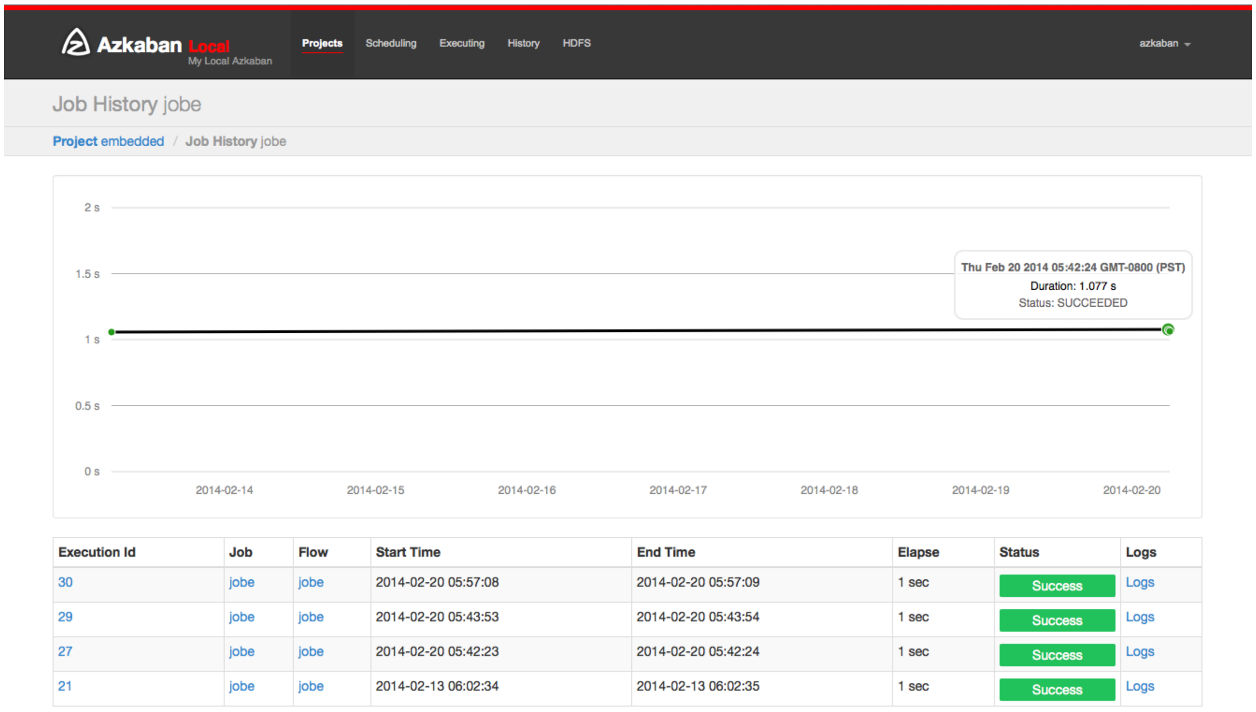
General Job Settings

Be Aware: A job may be shared by multiple flows. The change will be global!

Name	Value
fail	false
seconds	1
java.class	azkaban.test.executor.SleepJavaJob

Job History

Any retries of a job will show as `executionid.attempt` number.



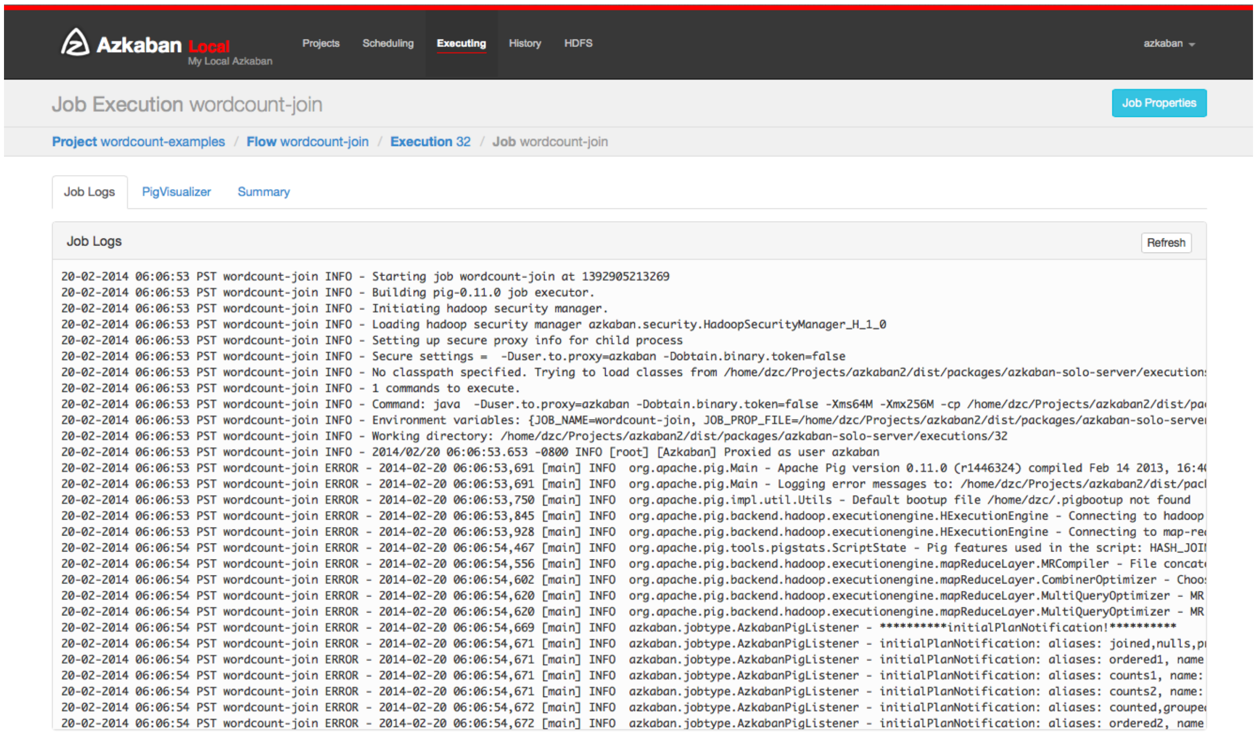
Execution Id	Job	Flow	Start Time	End Time	Elapse	Status	Logs
30	jobe	jobe	2014-02-20 05:57:08	2014-02-20 05:57:09	1 sec	Success	Logs
29	jobe	jobe	2014-02-20 05:43:53	2014-02-20 05:43:54	1 sec	Success	Logs
27	jobe	jobe	2014-02-20 05:42:23	2014-02-20 05:42:24	1 sec	Success	Logs
21	jobe	jobe	2014-02-13 06:02:34	2014-02-13 06:02:35	1 sec	Success	Logs

2.3.9 Job Details

From an execution page, after clicking “Job List” and then “Details” for one of the jobs, you will arrive at the job details page. This page contains tabs for the “Job Logs” and a “Summary”.

Job Logs

The job logs are stored in the database. They contain all the stdout and stderr output of the job.

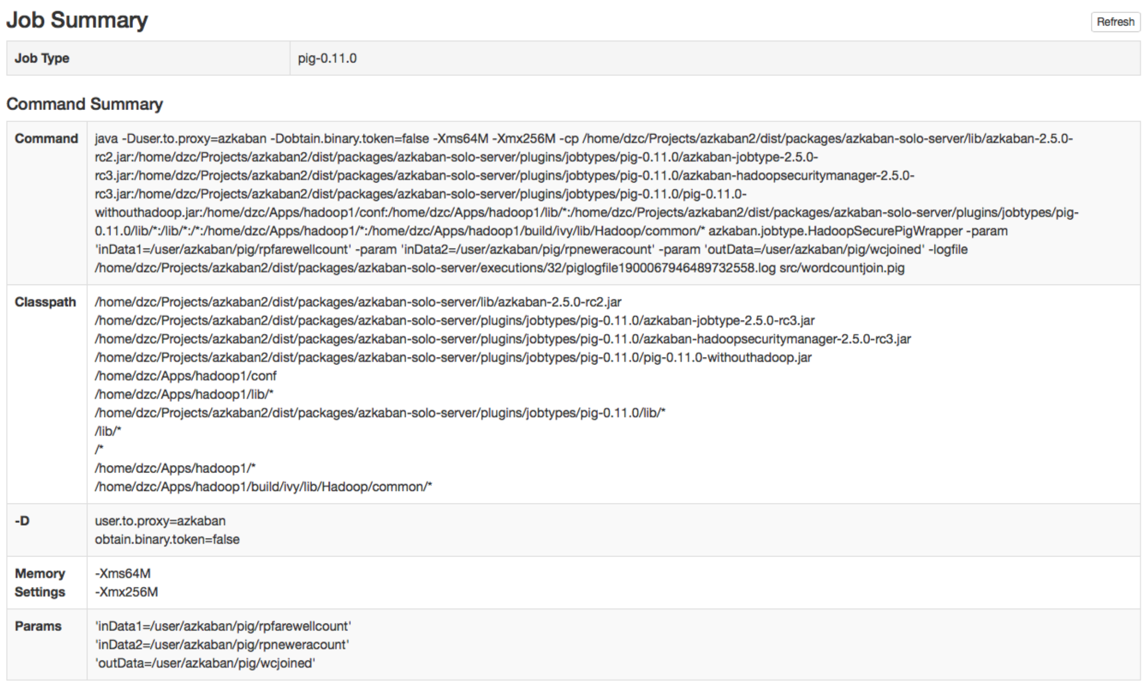


```
20-02-2014 06:06:53 PST wordcount-join INFO - Starting job wordcount-join at 1392905213269
20-02-2014 06:06:53 PST wordcount-join INFO - Building pig-0.11.0 job executor.
20-02-2014 06:06:53 PST wordcount-join INFO - Initiating hadoop security manager.
20-02-2014 06:06:53 PST wordcount-join INFO - Loading hadoop security manager azkaban.security.HadoopSecurityManager_H_1_0
20-02-2014 06:06:53 PST wordcount-join INFO - Setting up secure proxy info for child process
20-02-2014 06:06:53 PST wordcount-join INFO - Secure settings =  -Duser.to.proxy=azkaban -Dobtain.binary.token=false
20-02-2014 06:06:53 PST wordcount-join INFO - No classpath specified. Trying to load classes from /home/dzc/Projects/azkaban2/dist/packages/azkaban-solo-server/execution
20-02-2014 06:06:53 PST wordcount-join INFO - 1 commands to execute.
20-02-2014 06:06:53 PST wordcount-join INFO - Command: java  -Duser.to.proxy=azkaban -Dobtain.binary.token=false -Xms64M -Xmx256M -cp /home/dzc/Projects/azkaban2/dist/pa
20-02-2014 06:06:53 PST wordcount-join INFO - Environment variables: {JOB_NAME=wordcount-join, JOB_PROP_FILE=/home/dzc/Projects/azkaban2/dist/packages/azkaban-solo-serve
20-02-2014 06:06:53 PST wordcount-join INFO - Working directory: /home/dzc/Projects/azkaban2/dist/packages/azkaban-solo-server/executions/32
20-02-2014 06:06:53 PST wordcount-join INFO - 2014/02/20 06:06:53.653 -0800 INFO [root] [Azkaban] Proxied as user azkaban
20-02-2014 06:06:53 PST wordcount-join ERROR - 2014-02-20 06:06:53,691 [main] INFO  org.apache.pig.Main - Apache Pig version 0.11.0 (r1446324) compiled Feb 14 2013, 16:4
20-02-2014 06:06:53 PST wordcount-join ERROR - 2014-02-20 06:06:53,691 [main] INFO  org.apache.pig.Main - Logging error messages to: /home/dzc/Projects/azkaban2/dist/pac
20-02-2014 06:06:53 PST wordcount-join ERROR - 2014-02-20 06:06:53,750 [main] INFO  org.apache.pig.impl.util.Utils - Default bootup file /home/dzc/.pigbootup not found
20-02-2014 06:06:53 PST wordcount-join ERROR - 2014-02-20 06:06:53,845 [main] INFO  org.apache.pig.backend.hadoop.executionengine.HExecutionEngine - Connecting to hadoop
20-02-2014 06:06:53 PST wordcount-join ERROR - 2014-02-20 06:06:53,928 [main] INFO  org.apache.pig.backend.hadoop.executionengine.HExecutionEngine - Connecting to map-re
20-02-2014 06:06:54 PST wordcount-join ERROR - 2014-02-20 06:06:54,467 [main] INFO  org.apache.pig.tools.pigstats.ScriptState - Pig features used in the script: HASH_JOI
20-02-2014 06:06:54 PST wordcount-join ERROR - 2014-02-20 06:06:54,556 [main] INFO  org.apache.pig.backend.hadoop.executionengine.mapReduceLayer.MRCompiler - File concat
20-02-2014 06:06:54 PST wordcount-join ERROR - 2014-02-20 06:06:54,602 [main] INFO  org.apache.pig.backend.hadoop.executionengine.mapReduceLayer.CombinerOptimizer - Choo
20-02-2014 06:06:54 PST wordcount-join ERROR - 2014-02-20 06:06:54,620 [main] INFO  org.apache.pig.backend.hadoop.executionengine.mapReduceLayer.MultiQueryOptimizer - MR
20-02-2014 06:06:54 PST wordcount-join ERROR - 2014-02-20 06:06:54,620 [main] INFO  org.apache.pig.backend.hadoop.executionengine.mapReduceLayer.MultiQueryOptimizer - MR
20-02-2014 06:06:54 PST wordcount-join ERROR - 2014-02-20 06:06:54,669 [main] INFO  azkaban.jobtype.AzkabanPigListener - **********initialPlanNotification!**********
20-02-2014 06:06:54 PST wordcount-join ERROR - 2014-02-20 06:06:54,671 [main] INFO  azkaban.jobtype.AzkabanPigListener - initialPlanNotification: aliases: joined,nulls,p
20-02-2014 06:06:54 PST wordcount-join ERROR - 2014-02-20 06:06:54,671 [main] INFO  azkaban.jobtype.AzkabanPigListener - initialPlanNotification: aliases: ordered1, name
20-02-2014 06:06:54 PST wordcount-join ERROR - 2014-02-20 06:06:54,671 [main] INFO  azkaban.jobtype.AzkabanPigListener - initialPlanNotification: aliases: counts1, name:
20-02-2014 06:06:54 PST wordcount-join ERROR - 2014-02-20 06:06:54,671 [main] INFO  azkaban.jobtype.AzkabanPigListener - initialPlanNotification: aliases: counts2, name:
20-02-2014 06:06:54 PST wordcount-join ERROR - 2014-02-20 06:06:54,672 [main] INFO  azkaban.jobtype.AzkabanPigListener - initialPlanNotification: aliases: counted,groupe
20-02-2014 06:06:54 PST wordcount-join ERROR - 2014-02-20 06:06:54,672 [main] INFO  azkaban.jobtype.AzkabanPigListener - initialPlanNotification: aliases: ordered2, name
```

Job Summary

The Job Summary tab contains a summary of the information in the job logs. This includes:

- **Job Type** - the jobtype of the job
- **Command Summary** - the command that launched the job process, with fields such as the classpath and memory settings shown separately as well
- **Pig/Hive Job Summary** - custom stats specific to Pig and Hive jobs
- **Map Reduce Jobs** - a list of job ids of Map-Reduce jobs that were launched, linked to their job tracker pages



Job Summary

Refresh

Job Type	pig-0.11.0

Command Summary

Command	java -Duser.to.proxy=azkaban -Dobtain.binary.token=false -Xms64M -Xmx256M -cp /home/dzc/Projects/azkaban2/dist/packages/azkaban-solo-server/lib/azkaban-2.5.0-rc2.jar:/home/dzc/Projects/azkaban2/dist/packages/azkaban-solo-server/plugins/jobtypes/pig-0.11.0/azkaban-jobtype-2.5.0-rc3.jar:/home/dzc/Projects/azkaban2/dist/packages/azkaban-solo-server/plugins/jobtypes/pig-0.11.0/azkaban-hadoopsecuritymanager-2.5.0-rc3.jar:/home/dzc/Projects/azkaban2/dist/packages/azkaban-solo-server/plugins/jobtypes/pig-0.11.0/pig-0.11.0-withouthadoop.jar:/home/dzc/Apps/hadoop1/conf:/home/dzc/Apps/hadoop1/lib/*:/home/dzc/Projects/azkaban2/dist/packages/azkaban-solo-server/plugins/jobtypes/pig-0.11.0/lib/*:/lib/*:/*:/home/dzc/Apps/hadoop1/*:/home/dzc/Apps/hadoop1/build/ivy/lib/Hadoop/common/* azkaban.jobtype.HadoopSecurePigWrapper -param 'inData1=/user/azkaban/pig/rpfarewellcount' -param 'inData2=/user/azkaban/pig/rpneweracount' -param 'outData=/user/azkaban/pig/wcjoined' -logfile /home/dzc/Projects/azkaban2/dist/packages/azkaban-solo-server/executions/32/piglogfile1900067946489732558.log src/wordcountjoin.pig
Classpath	/home/dzc/Projects/azkaban2/dist/packages/azkaban-solo-server/lib/azkaban-2.5.0-rc2.jar /home/dzc/Projects/azkaban2/dist/packages/azkaban-solo-server/plugins/jobtypes/pig-0.11.0/azkaban-jobtype-2.5.0-rc3.jar /home/dzc/Projects/azkaban2/dist/packages/azkaban-solo-server/plugins/jobtypes/pig-0.11.0/azkaban-hadoopsecuritymanager-2.5.0-rc3.jar /home/dzc/Projects/azkaban2/dist/packages/azkaban-solo-server/plugins/jobtypes/pig-0.11.0/pig-0.11.0-withouthadoop.jar /home/dzc/Apps/hadoop1/conf /home/dzc/Apps/hadoop1/lib/* /home/dzc/Projects/azkaban2/dist/packages/azkaban-solo-server/plugins/jobtypes/pig-0.11.0/lib/* /lib/* /* /home/dzc/Apps/hadoop1/* /home/dzc/Apps/hadoop1/build/ivy/lib/Hadoop/common/*

-D	user.to.proxy=azkaban
	obtain.binary.token=false

Memory Settings	-Xms64M -Xmx256M
Params	'inData1=/user/azkaban/pig/rpfarewellcount' 'inData2=/user/azkaban/pig/rpneweracount' 'outData=/user/azkaban/pig/wcjoined'

2.4 Flow Trigger Dependency Plugin

2.4.1 Event Based Trigger

Currently there are only few ways to launch jobs in Azkaban including schedules and API. However, they are limited because sometimes jobs need to be executed automatically on demand. Event trigger is a new feature introduced by Azkaban. It defines a new paradigm of triggering flows - triggering a flow on event arrival. This concept enables users to define events that the flow depends on. Once all of the dependencies become ready, a workflow will be triggered.

Apache Kafka is a Publish & Subscribe data streaming system. By utilizing Kafka, we do the regular expression match on the Kafka event payload. With the contain-a logic matching, a dependency will be marked as satisfied only if the whole payload contains the Regex pattern that user predefines.

2.4.2 Getting started with Deploying Event Trigger on the Azkaban

Azkaban builds use Gradle (downloads automatically when run using gradlew which is the Gradle wrapper) and requires Java 8 or higher.

Build

The following commands run on *nix platforms like Linux, OS X. For building Flow Trigger Dependency Plugin, we need to run the comment in `/az-flow-trigger-dependency-type/kafka-event-trigger` directory.

```
# Build Azkaban
../../gradlew build

# Clean the build
```

(continues on next page)

(continued from previous page)

```
../../gradlew clean

# Build without running tests
../../gradlew build -x test
```

These are all standard Gradle commands. Please look at Gradle documentation for more info.

Server Configuration

The gradlew commands help you to build the fat JAR. After that, you need to specify the plugin.dir within `conf`. Take solo-server for example, override the `azkaban.dependency.plugin.dir` property for runtime parameters inside the `azkaban.properties` file under the solo-server `conf` directory. This property needs to set to contain the location where you put your Event-Trigger JAR file.

Data Base Configuration (Optional)

The following 4 properties can be defined in `conf/azkaban.private.properties`. for solo-server based on the use case.

Properties
mysql.user
mysql.password
org.quartz.dataSource.quartzDS.user
org.quartz.dataSource.quartzDS.password

2.4.3 Event Based Trigger Plugin Configuration

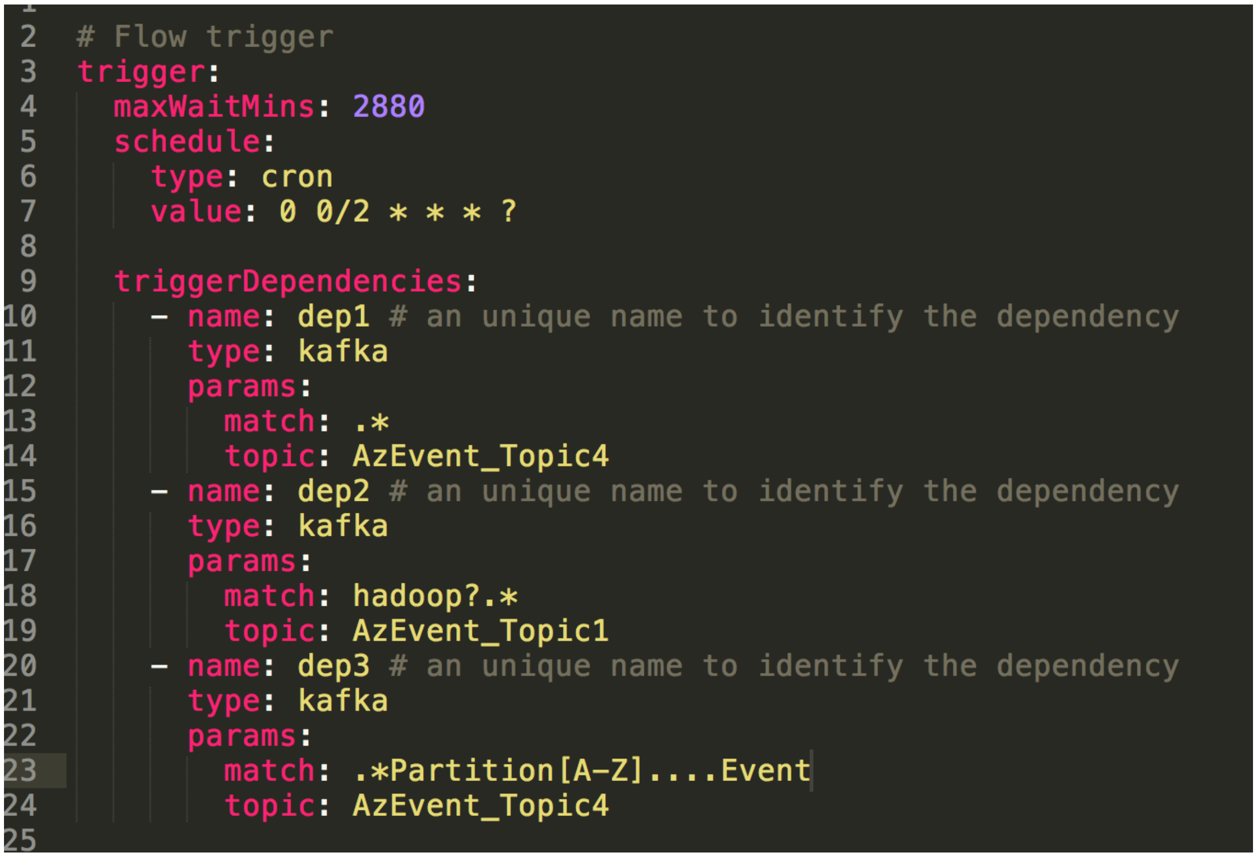
Inside the Azkaban dependency plugin directory, there should be two items Event Based Trigger plugin jar and the `dependency.properties`.

Required properties are:

- **dependency.classpath** - Used by Azkaban identify plugins classpath. Should be the JAR file's absolute path.
- **dependency.class** - Used by Azkaban flow trigger instance to integrate with this configuration file. Take Event trigger for example, it should be `trigger.kafka.KafkaDependencyCheck`.
- **kafka.broker.url** - Specifying URL and port number where your Kafka broker is.

2.4.4 Event Trigger Instance Configuration

Event trigger is part of flow definition and each flow can only have one event trigger at most. Defining an event trigger is supported via Hadoop DSL. The trigger needs to be configurated within the flow file along with the project zip that users upload. Event trigger is composed of a list of event dependencies, max wait time and schedule. Take the following figure as example:



```yaml

# Flow trigger
trigger:
  maxWaitMins: 2880
  schedule:
    type: cron
    value: 0 0/2 * * * ?

  triggerDependencies:
    - name: dep1 # an unique name to identify the dependency
      type: kafka
      params:
        match: .*
        topic: AzEvent_Topic4
    - name: dep2 # an unique name to identify the dependency
      type: kafka
      params:
        match: hadoop?.*
        topic: AzEvent_Topic1
    - name: dep3 # an unique name to identify the dependency
      type: kafka
      params:
        match: .*Partition[A-Z]....Event
        topic: AzEvent_Topic4

```

- **Max Wait Time**: How long the trigger will wait for all dependencies to be available before cancelling it.
- **Trigger.schedule**: The schedule to perform this workflow on the regular basis. We use the cron time format here to specify, creating a trigger followed by the project workflow every 2 minutes
- **Trigger.schedule**: The params here is to clarify what regex pattern happening in the event coming from specific topic channel. The trigger kick-starts the flow if all of predefined dependency conditions are met.

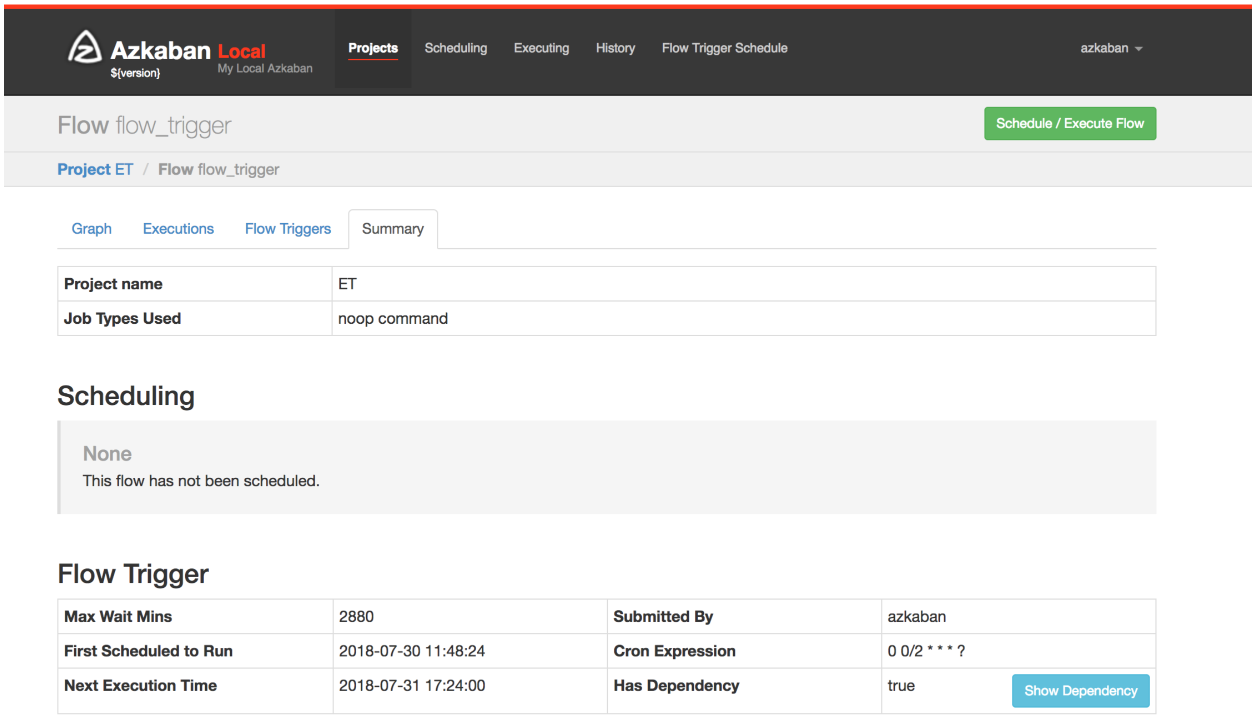
Therefore, this trigger example will launch the flow once detecting Kafka event with anything in `AzEvent_Topic4`, `.*Partition[A-Z]....Event` string in event comming from `AzEvent_Topic4` and `hadoop?.*` in `AzEvent_Topic1`.

The matching mechanism can be extended other than regex since now it is implemented as a generic interface.

2.4.5 Event Based Trigger Example With Azkaban UI

All scheduled data trigger will show up Azkaban Flow Trigger section. Also, project admins are able to pause and resume a scheduled trigger for undesirable situation.

Trigger info page for a specific flow:



Project name	ET
Job Types Used	noop command

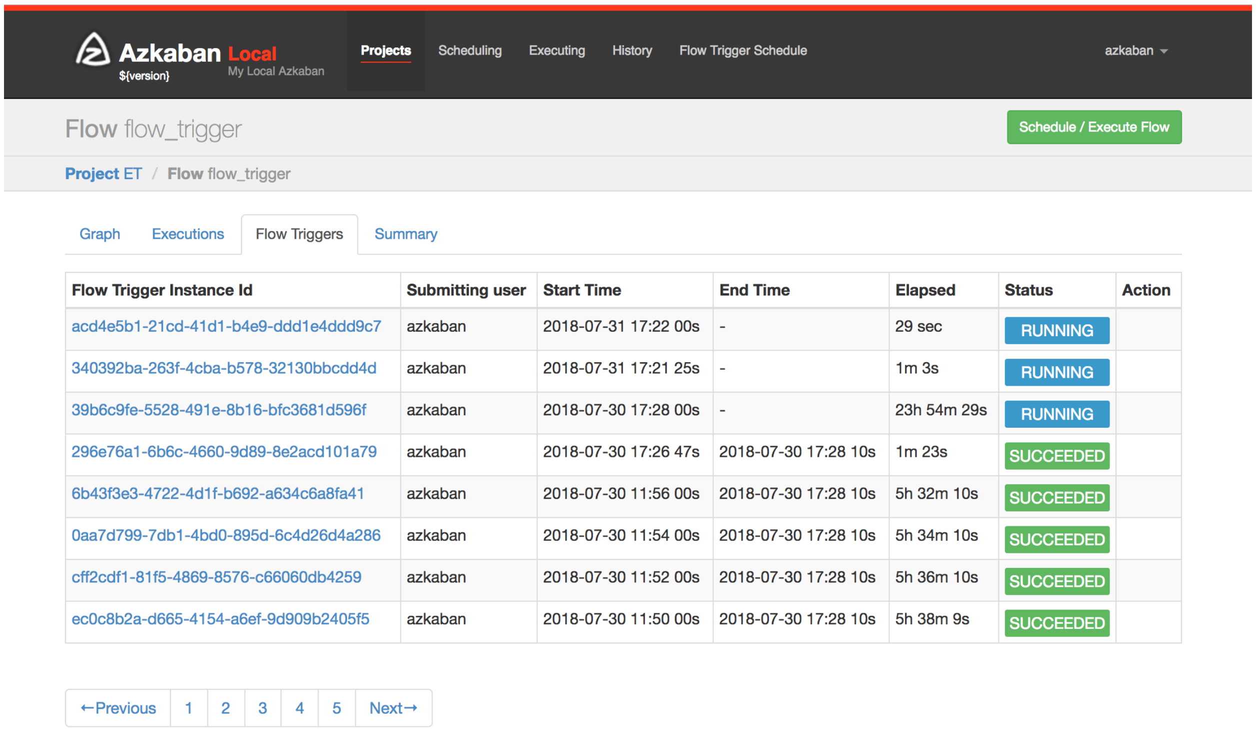
Scheduling

None
This flow has not been scheduled.

Flow Trigger

Max Wait Mins	2880	Submitted By	azkaban
First Scheduled to Run	2018-07-30 11:48:24	Cron Expression	0 0/2 * * * ?
Next Execution Time	2018-07-31 17:24:00	Has Dependency	true Show Dependency

Current and historic triggers for a specific flow:



Flow Trigger Instance Id	Submitting user	Start Time	End Time	Elapsed	Status	Action
acd4e5b1-21cd-41d1-b4e9-ddd1e4ddd9c7	azkaban	2018-07-31 17:22 00s	-	29 sec	RUNNING	
340392ba-263f-4cba-b578-32130bbcdd4d	azkaban	2018-07-31 17:21 25s	-	1m 3s	RUNNING	
39b6c9fe-5528-491e-8b16-bfc3681d596f	azkaban	2018-07-30 17:28 00s	-	23h 54m 29s	RUNNING	
296e76a1-6b6c-4660-9d89-8e2acd101a79	azkaban	2018-07-30 17:26 47s	2018-07-30 17:28 10s	1m 23s	SUCCEEDED	
6b43f3e3-4722-4d1f-b692-a634c6a8fa41	azkaban	2018-07-30 11:56 00s	2018-07-30 17:28 10s	5h 32m 10s	SUCCEEDED	
0aa7d799-7db1-4bd0-895d-6c4d26d4a286	azkaban	2018-07-30 11:54 00s	2018-07-30 17:28 10s	5h 34m 10s	SUCCEEDED	
cff2cdf1-81f5-4869-8576-c66060db4259	azkaban	2018-07-30 11:52 00s	2018-07-30 17:28 10s	5h 36m 10s	SUCCEEDED	
ec0c8b2a-d665-4154-a6ef-9d909b2405f5	azkaban	2018-07-30 11:50 00s	2018-07-30 17:28 10s	5h 38m 9s	SUCCEEDED	

Follow these steps to run end to end local test:

1.Start Kafka Broker Locally:

Following Kafka QuickStart to run these Kafka console scripts in Kafka package

```
*Start ZooKeeper
bin/zookeeper-server-start.sh config/zookeeper.properties
*Start Kafka Server
bin/kafka-server-start.sh config/server.properties
```

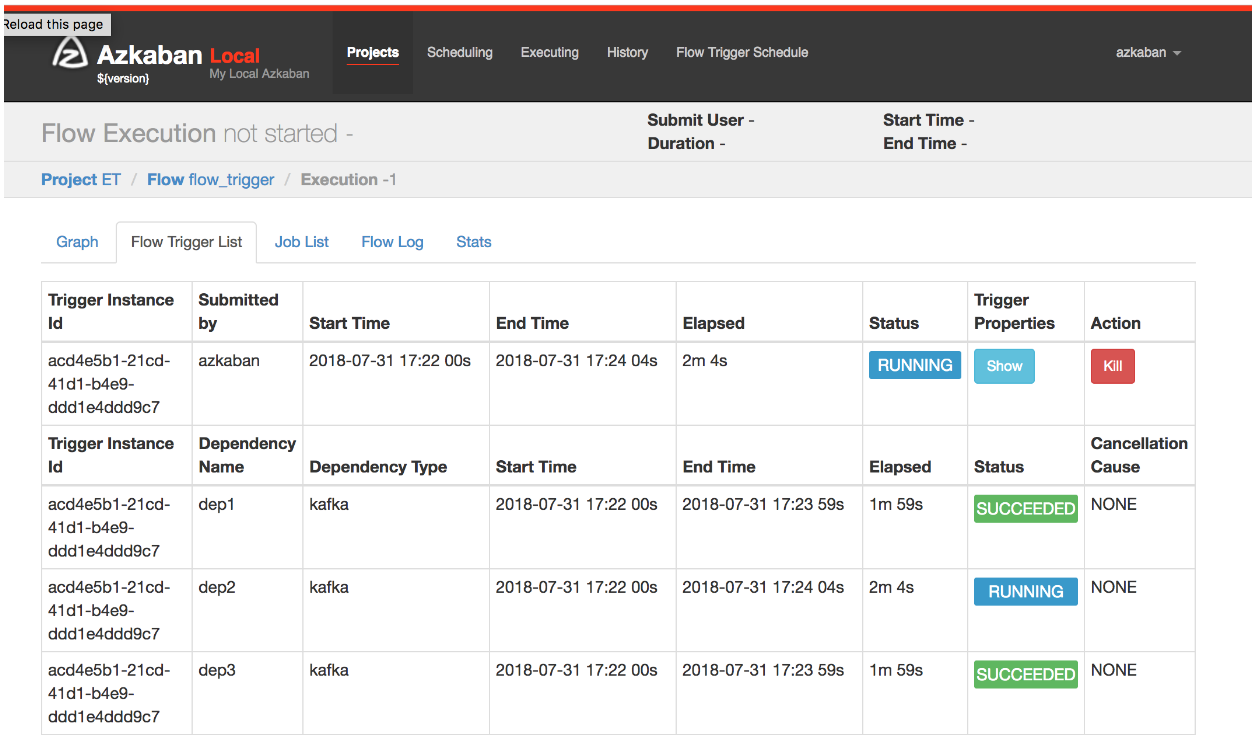
2. Send Json event to topics with AzEvent_Topic4 as example:

```
bin/kafka-console-producer.sh --broker-list localhost:9092 --topic AzEvent_Topic4 <␣
↪recordPartition.json
```

Here is how my `recordPartition.json` looks like:

```
{
    "name":"Charlie",
    "team": "Azkaban",
    "event":"MetastorePartitionAuditEvent"
}
```

Once this event arrived, Azkaban will mark this specific event dependency as success.



Flow Execution not started -

Submit User -
Duration -
Start Time -
End Time -

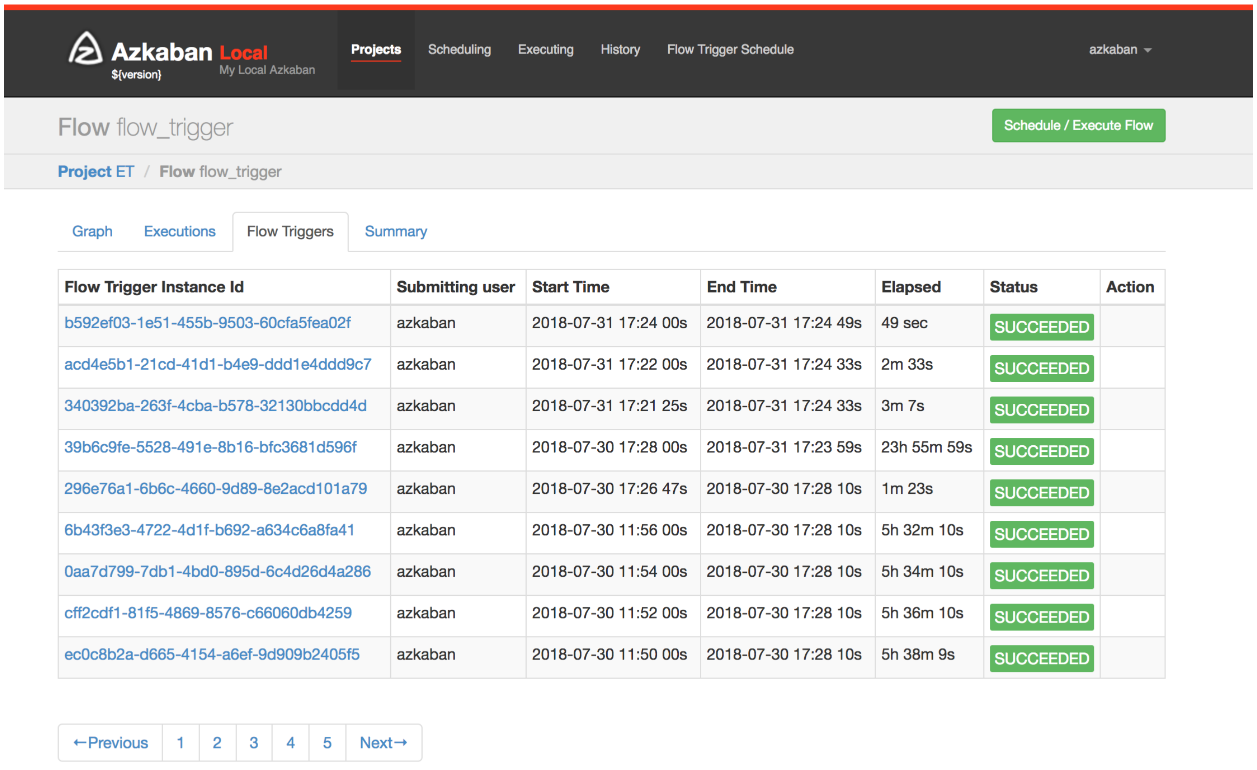
Project ET / Flow flow_trigger / Execution -1

Trigger Instance Id	Submitted by	Start Time	End Time	Elapsed	Status	Trigger Properties	Action
acd4e5b1-21cd-41d1-b4e9-ddd1e4ddd9c7	azkaban	2018-07-31 17:22 00s	2018-07-31 17:24 04s	2m 4s	RUNNING	Show	Kill
Trigger Instance Id	**Dependency Name**	**Dependency Type**	**Start Time**	**End Time**	**Elapsed**	**Status**	**Cancellation Cause**
acd4e5b1-21cd-41d1-b4e9-ddd1e4ddd9c7	dep1	kafka	2018-07-31 17:22 00s	2018-07-31 17:23 59s	1m 59s	SUCCEEDED	NONE
acd4e5b1-21cd-41d1-b4e9-ddd1e4ddd9c7	dep2	kafka	2018-07-31 17:22 00s	2018-07-31 17:24 04s	2m 4s	RUNNING	NONE
acd4e5b1-21cd-41d1-b4e9-ddd1e4ddd9c7	dep3	kafka	2018-07-31 17:22 00s	2018-07-31 17:23 59s	1m 59s	SUCCEEDED	NONE

3. Send another event from producer to launch the flow:

```
bin/kafka-console-producer.sh --broker-list localhost:9092 --topic AzEvent_Topic4 <␣
↪recordHadoop.json
```

Trigger the workflows that have all dependencies cleared out:



Flow flow_trigger

Project ET / Flow flow_trigger

Flow Trigger Instance Id	Submitting user	Start Time	End Time	Elapsed	Status	Action
b592ef03-1e51-455b-9503-60cfa5fea02f	azkaban	2018-07-31 17:24 00s	2018-07-31 17:24 49s	49 sec	SUCCEEDED	
acd4e5b1-21cd-41d1-b4e9-ddd1e4ddd9c7	azkaban	2018-07-31 17:22 00s	2018-07-31 17:24 33s	2m 33s	SUCCEEDED	
340392ba-263f-4cba-b578-32130bbcdd4d	azkaban	2018-07-31 17:21 25s	2018-07-31 17:24 33s	3m 7s	SUCCEEDED	
39b6c9fe-5528-491e-8b16-bfc3681d596f	azkaban	2018-07-30 17:28 00s	2018-07-31 17:23 59s	23h 55m 59s	SUCCEEDED	
296e76a1-6b6c-4660-9d89-8e2acd101a79	azkaban	2018-07-30 17:26 47s	2018-07-30 17:28 10s	1m 23s	SUCCEEDED	
6b43f3e3-4722-4d1f-b692-a634c6a8fa41	azkaban	2018-07-30 11:56 00s	2018-07-30 17:28 10s	5h 32m 10s	SUCCEEDED	
0aa7d799-7db1-4bd0-895d-6c4d26d4a286	azkaban	2018-07-30 11:54 00s	2018-07-30 17:28 10s	5h 34m 10s	SUCCEEDED	
cff2cdf1-81f5-4869-8576-c66060db4259	azkaban	2018-07-30 11:52 00s	2018-07-30 17:28 10s	5h 36m 10s	SUCCEEDED	
ec0c8b2a-d665-4154-a6ef-9d909b2405f5	azkaban	2018-07-30 11:50 00s	2018-07-30 17:28 10s	5h 38m 9s	SUCCEEDED	

2.4.6 Limitation

Since our design purpose is to decouple the trigger condition from the action to take, currently there is a limitation on deserializing record. Although Kafka provides the ability to publish and subscribe to streams of records on custom serializer and deserializer. What we have right now is limited to Kafka build in String deserializer only. We are planning to enhance the flexibility on users to upload JAR with their own custom deserialize function in the near future.

CHAPTER 3

Community

Users and development team are in the Gitter.

Developers interested in getting more involved with Azkaban may join the mailing lists mailing lists, report bugs, and make contributions.