

工作流调度器 azkaban

为什么需要工作流调度系统

- 一个完整的数据分析系统通常都是由大量任务单元组成：
shell 脚本程序，java 程序，mapreduce 程序、hive 脚本等
- 各任务单元之间存在时间先后及前后依赖关系
- 为了很好地组织起这样的复杂执行计划，需要一个工作流调度系统来调度执行；

例如，我们可能有这样一个需求，某个业务系统每天产生 20G 原始数据，我们每天都要对其进行处理，处理步骤如下所示：

- 1、通过 Hadoop 先将原始数据同步到 HDFS 上；
- 2、借助 MapReduce 计算框架对原始数据进行转换，生成的数据以分区表的形式存储到多张 Hive 表中；
- 3、需要对 Hive 中多个表的数据进行 JOIN 处理，得到一个明细数据 Hive 大表；
- 4、将明细数据进行各种统计分析，得到结果报表信息；
- 5、需要将统计分析得到的结果数据同步到业务系统中，供业务调用使用。

工作流调度实现方式

简单的任务调度：直接使用 linux 的 crontab 来定义；

复杂的任务调度：开发调度平台或使用现成的开源调度系统，比如 oozie、azkaban、airflow 等

常见工作流调度系统

市面上目前有许多工作流调度器

在 hadoop 领域，常见的工作流调度器有 Oozie, Azkaban, Cascading, Hamake 等

2Azkaban 介绍

概述

azkaban 官网：

<https://azkaban.github.io/>

Azkaban 是由 LinkedIn 开源的一个批量工作流任务调度器。用于在一个工作流内以一个特定的顺序运行一组工作和流程。

Azkaban 定义了一种 KV 文件(properties)格式来建立任务之间的依赖关系，并提供一个易于使用的 web 用户界面维护和跟踪你的工作流。

它有如下功能特点：

- ✧ Web 用户界面
- ✧ 方便上传工作流
- ✧ 方便设置任务之间的关系
- ✧ 调度工作流
- ✧ 认证/授权(权限的工作)
- ✧ 能够杀死并重新启动工作流
- ✧ 模块化和可插拔的插件机制
- ✧ 项目工作区
- ✧ 工作流和任务的日志记录和审计

Azkaban 安装部署

2.3.2、azkaban 单服务模式安装与使用

所需软件

azkaban-solo-server

单服务模式安装

第一步：解压

azkaban 的 solo server 使用的是一个单节点的模式来进行启动服务的，只需要一个 azkaban-solo-server-0.1.0-SNAPSHOT.tar.gz 的安装包即可启动，所有的数据信息都是保存在 H2 这个 azkaban 默认的数据当中，上传我们的压缩包，然后修改配置文件启动即可

```
cd /export/softwares
tar -zxvf azkaban-solo-server-0.1.0-SNAPSHOT.tar.gz -C ../servers/
```

第二步：修改两个配置文件

修改时区配置文件

```
cd /export/servers/azkaban-solo-server-0.1.0-SNAPSHOT/conf
vim azkaban.properties
```

```
default.timezone.id=Asia/Shanghai
```

```
# Azkaban Personalization Settings
azkaban.name=Slg1705 看你心情,随便,任意
azkaban.label=My Local Azkaban
azkaban.color=#FF3601
azkaban.default.servlet.path=/index
web.resource.dir=web/
default.timezone.id=Asia/Shanghai
# Azkaban UserManager class
user.manager.class=azkaban.user.XmlUserManager
user.manager.xml.file=conf/azkaban-users.xml
# Loader for projects
executor.global.properties=conf/global.properties
azkaban.project.dir=projects
database.type=h2
h2.path=./h2
h2.create.tables=true
# Velocity dev mode
velocity.dev.mode=false
# Azkaban Jetty server properties.
jetty.use.ssl=false
jetty.maxThreads=25
jetty.port=8081
# Azkaban Executor settings
executor.port=12321
# mail settings
mail.sender=
mail.host=
# User facing web server configurations used to construct
a reverse proxy between Azkaban web servers and users.
```

修改 commonprivate.properties 配置文件

```
cd /export/servers/azkaban-solo-server-0.1.0-SNAPSHOT/plugins/jobtypes
vim commonprivate.properties
```

```
execute.as.user=false
memCheck.enabled=false
```

```
# set execute-as-user
execute.as.user=false
memCheck.enabled=false
~
~ 添加这一行,避免内存检查
```

第三步：启动 solo-server

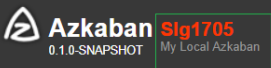
启动 azkaban-solo-server

```
cd /export/servers/azkaban-solo-server-0.1.0-SNAPSHOT  
bin/start-solo.sh
```

```
[root@node03 jobtypes]# cd /export/servers/azkaban-solo-server-0.1.0-SNAPSHOT  
[root@node03 azkaban-solo-server-0.1.0-SNAPSHOT]# bin/start-solo.sh  
[root@node03 azkaban-solo-server-0.1.0-SNAPSHOT]# jps  
5504 AzkabanSingleServer  
2599 Kafka  
2538 QuorumPeerMain  
3053 Application  
2110 NodeManager  
5534 Jps  
1999 DataNode
```

第四步：浏览器页面访问

浏览器页面访问 <http://node03:8081/>



My Local Azkaban

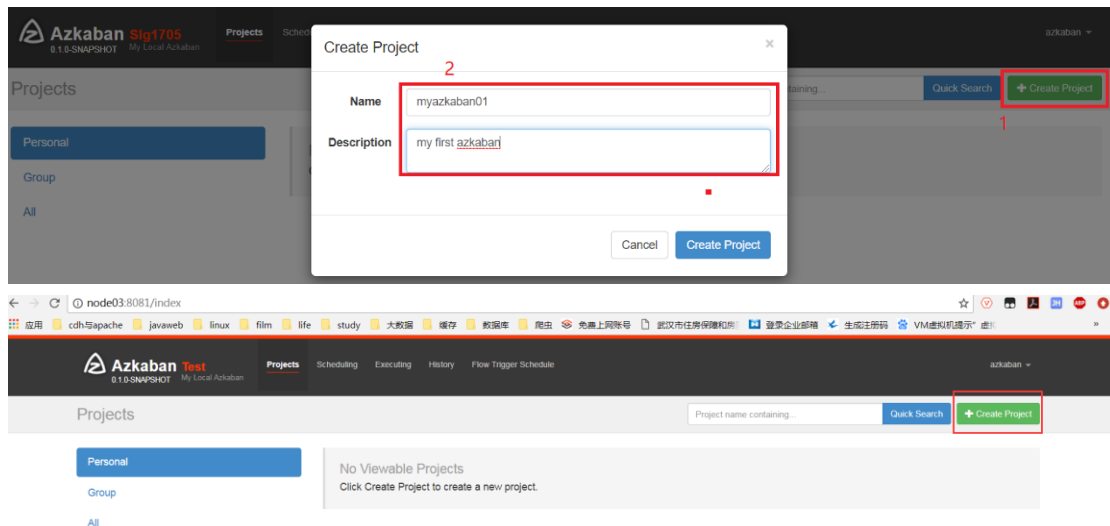
Login

Username	默认用户名密码都是
Password	azkaban

Login

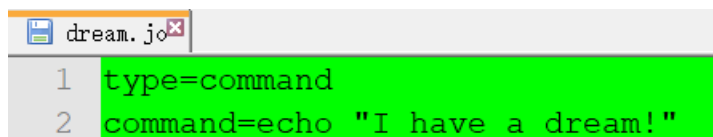
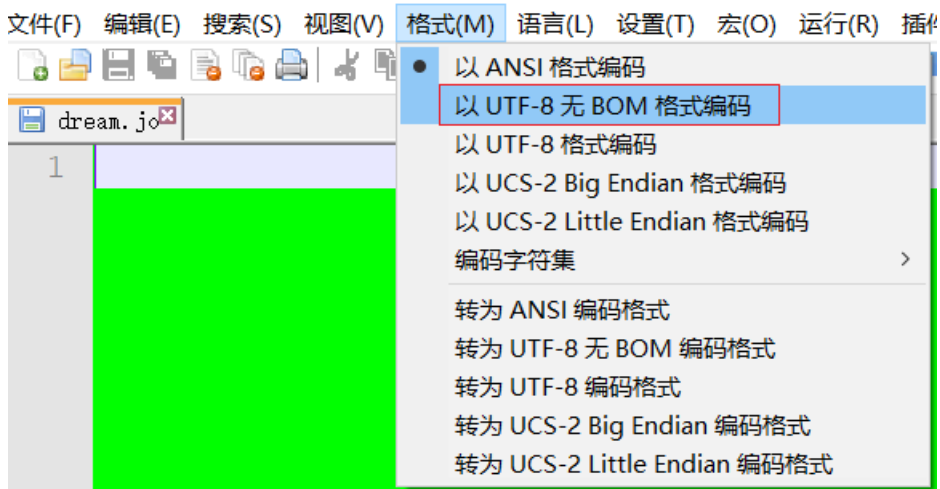
单服务模式使用

需求：使用 azkaban 调度我们的 shell 脚本，执行 linux 的 shell 命令

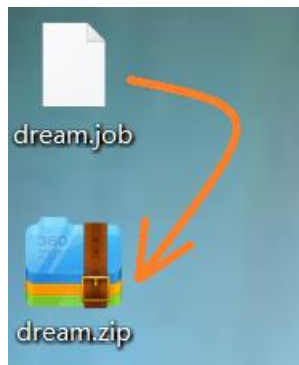


创建普通文本文件 dream.job，文件内容如下

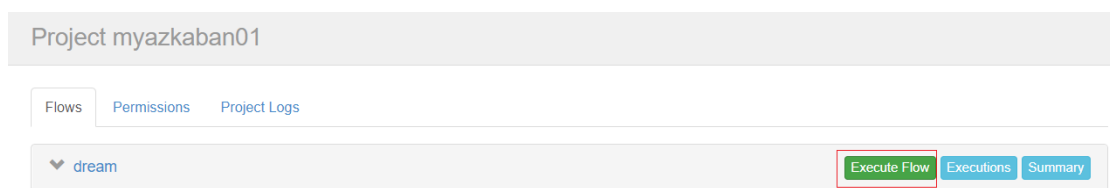
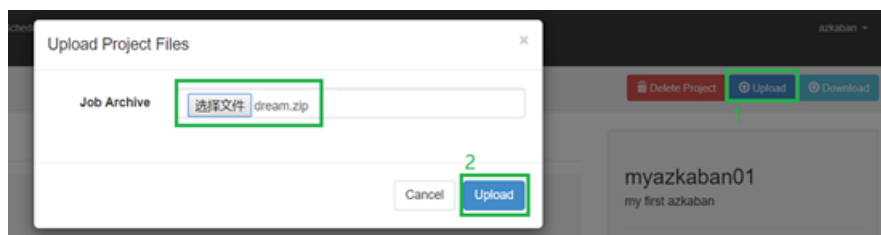
```
type=command
command=echo "I have a dream!"
```



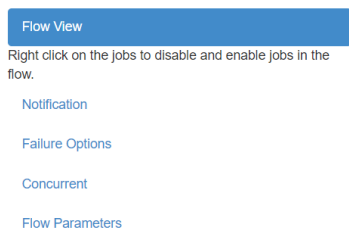
然后将这个文件打包为压缩文件，如下：



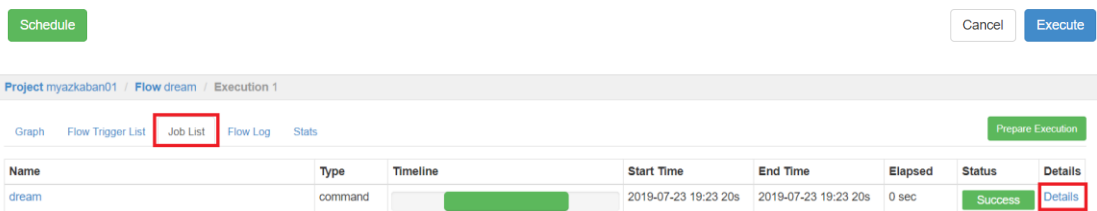
azkaban 上传我们的压缩包

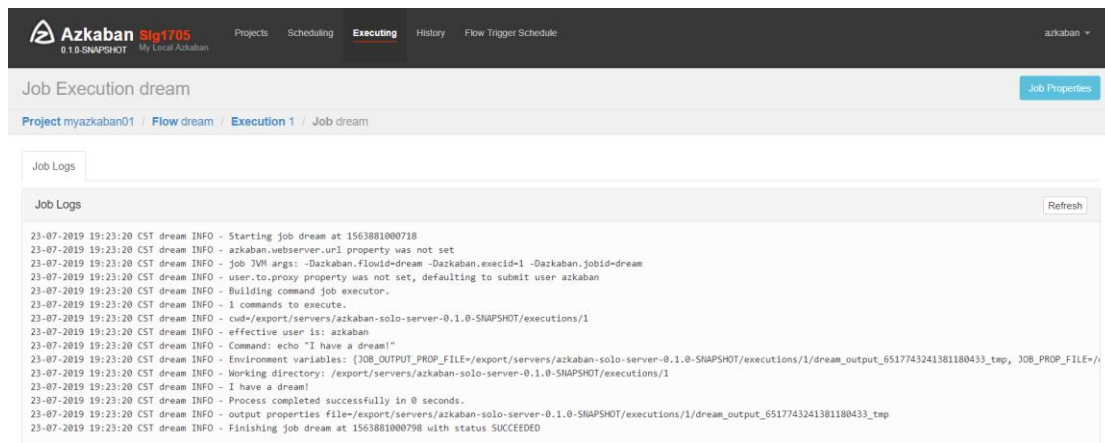


Execute Flow dream



dream





2.3.3、azkaban 两个服务模式安装

1、确认所需软件：

Azkaban Web 服务安装包

azkaban-web-server-0.1.0-SNAPSHOT.tar.gz

Azkaban 执行服务安装包

azkaban-exec-server-0.1.0-SNAPSHOT.tar.gz

编译之后的 sql 脚本

create-all-sql-0.1.0-SNAPSHOT.sql

C 程序文件脚本

execute-as-user.c 程序

2、数据库准备

进入 mysql 的客户端执行以下命令

```
mysql -uroot -p
```

执行以下命令：

```
CREATE DATABASE azkaban;  
CREATE USER 'azkaban'@'%' IDENTIFIED BY 'azkaban';  
GRANT all privileges ON azkaban.* to 'azkaban'@'%' identified by  
'azkaban' WITH GRANT OPTION;  
flush privileges;
```

```
use azkaban;  
source /export/softwares/create-all-sql-0.1.0-SNAPSHOT.sql;
```

3、解压软件安装包

解压 azkaban-web-server

```
cd /export/softwares  
tar -zxvf azkaban-web-server-0.1.0-SNAPSHOT.tar.gz -C ../servers/  
cd /export/servers  
mv azkaban-web-server-0.1.0-SNAPSHOT/ azkaban-web-server-3.51.0
```

解压 azkaban-exec-server

```
cd /export/softwares  
tar -zxvf azkaban-exec-server-0.1.0-SNAPSHOT.tar.gz -C ../servers/  
cd /export/servers  
mv azkaban-exec-server-0.1.0-SNAPSHOT/ azkaban-exec-server-3.51.0
```

4、安装 SSL 安全认证 允许我们使用 https 的方式访问 azkaban 的 web 服务

密码 **azkaban** 一定要一个个的字母输入，或者粘贴也行

```
cd /export/servers/azkaban-web-server-3.51.0  
keytool -keystore keystore -alias jetty -genkey -keyalg RSA
```


5、azkaban web server 安装

修改 azkaban-web-server 的配置文件

```
cd /export/servers/azkaban-web-server-3.51.0/conf  
vim azkaban.properties
```

```
# Azkaban Personalization Settings  
azkaban.name=Azkaban  
azkaban.label=My Azkaban  
azkaban.color=#FF3601  
azkaban.default.servlet.path=/index  
web.resource.dir=web/  
default.timezone.id=Asia/Shanghai  
  
# Azkaban UserManager class  
user.manager.class=azkaban.user.XmlUserManager  
user.manager.xml.file=conf/azkaban-users.xml  
  
# Loader for projects  
executor.global.properties=conf/global.properties  
azkaban.project.dir=projects  
  
# Velocity dev mode  
velocity.dev.mode=false  
  
# Azkaban Jetty server properties.  
jetty.use.ssl=true  
jetty.maxThreads=25  
jetty.port=8081  
  
jetty.keystore=/export/servers/azkaban-web-server-3.51.0/keystore  
jetty.password=azkaban  
jetty.keypassword=azkaban  
jetty.truststore=/export/servers/azkaban-web-server-3.51.0/keystore  
jetty.trustpassword=azkaban  
  
# Azkaban Executor settings
```

```
# mail settings
mail.sender=
mail.host=
# User facing web server configurations used to construct the user facing server
URLs. They are useful when there is a reverse proxy between Azkaban web
servers and users.
# enduser -> myazkabanhost:443 -> proxy -> localhost:8081
# when this parameters set then these parameters are used to generate email
links.
# if these parameters are not set then jetty.hostname, and jetty.port(if ssl
configured jetty.ssl.port) are used.
# azkaban.webserver.external_hostname=myazkabanhost.com
# azkaban.webserver.external_ssl_port=443
# azkaban.webserver.external_port=8081
job.failure.email=
job.success.email=
lockdown.create.projects=false
cache.directory=cache
# JMX stats
jetty.connector.stats=true
executor.connector.stats=true
# Azkaban mysql settings by default. Users should configure their own
username and password.
database.type=mysql
mysql.port=3306
mysql.host=node03
mysql.database=azkaban
mysql.user=azkaban
mysql.password=azkaban
mysql.numconnections=100
#Multiple Executor
azkaban.use.multiple.executors=true
#azkaban.executorselector.filters=StaticRemainingFlowSize,MinimumFreeMe
memory,CpuStatus
azkaban.executorselector.comparator.NumberOfAssignedFlowComparator=1
azkaban.executorselector.comparator.Memory=1
azkaban.executorselector.comparator.LastDispatched=1
azkaban.executorselector.comparator.CpuUsage=1
```

```
#executor.port=12321
```

6、azkaban executor server 安装

第一步：修改 azkaban-exec-server 配置文件

修改 azkaban-exec-server 的配置文件

```
cd /export/servers/azkaban-exec-server-3.51.0/conf  
vim azkaban.properties
```

```
# Azkaban Personalization Settings  
azkaban.name=Azkaban  
azkaban.label=My Azkaban  
azkaban.color=#FF3601  
azkaban.default.servlet.path=/index  
web.resource.dir=web/  
default.timezone.id=Asia/Shanghai  
  
# Azkaban UserManager class  
user.manager.class=azkaban.user.XmlUserManager  
user.manager.xml.file=conf/azkaban-users.xml  
  
# Loader for projects  
executor.global.properties=conf/global.properties  
azkaban.project.dir=projects  
  
# Velocity dev mode  
velocity.dev.mode=false  
  
# Azkaban Jetty server properties.  
jetty.use.ssl=true  
jetty.maxThreads=25  
jetty.port=8081  
  
jetty.keystore=/export/servers/azkaban-web-server-3.51.0/keystore  
jetty.password=azkaban  
jetty.keypassword=azkaban  
jetty.truststore=/export/servers/azkaban-web-server-3.51.0/keystore  
jetty.trustpassword=azkaban
```

```
# Where the Azkaban web server is located
azkaban.webserver.url=https://node03:8443
# mail settings
mail.sender=
mail.host=
# User facing web server configurations used to construct the user facing server
URLs. They are useful when there is a reverse proxy between Azkaban web
servers and users.
# enduser -> myazkabanhost:443 -> proxy -> localhost:8081
# when this parameters set then these parameters are used to generate email
links.
# if these parameters are not set then jetty.hostname, and jetty.port(if ssl
configured jetty.ssl.port) are used.
# azkaban.webserver.external_hostname=myazkabanhost.com
# azkaban.webserver.external_ssl_port=443
# azkaban.webserver.external_port=8081
job.failure.email=
job.success.email=
lockdown.create.projects=false
cache.directory=cache
# JMX stats
jetty.connector.stats=true
executor.connector.stats=true
# Azkaban plugin settings
azkaban.jobtype.plugin.dir=plugins/jobtypes
# Azkaban mysql settings by default. Users should configure their own
username and password.
database.type=mysql
mysql.port=3306
mysql.host=node03
mysql.database=azkaban
mysql.user=azkaban
mysql.password=azkaban
mysql.numconnections=100
# Azkaban Executor settings
executor.maxThreads=50
executor.flow.threads=30
```

第二步：添加插件

将我们编译后的 C 文件 `execute-as-user.c`

上传到这个目录来 `/export/servers/azkaban-exec-server-3.51.0/plugins/jobtypes`

或者直接将我们 `/export/softwares` 下面的文件拷贝过来也行

```
cp /export/softwares/execute-as-user.c /export/servers/azkaban-exec-server-3.51.0/plugins/jobtypes/
```

然后执行以下命令生成 `execute-as-user`

```
yum -y install gcc-c++  
cd /export/servers/azkaban-exec-server-3.51.0/plugins/jobtypes  
gcc execute-as-user.c -o execute-as-user  
chown root execute-as-user  
chmod 6050 execute-as-user
```

第三步：修改配置文件

修改配置文件

```
cd /export/servers/azkaban-exec-server-3.51.0/plugins/jobtypes  
vim commonprivate.properties
```

```
execute.as.user=false  
memCheck.enabled=false  
azkaban.native.lib=/export/servers/azkaban-exec-server-  
3.51.0/plugins/jobtypes
```

7、启动服务

第一步：启动 azkaban exec server

```
cd /export/servers/azkaban-exec-server-3.51.0  
bin/start-exec.sh
```

第二步：激活我们的 exec-server

node03 机器任意目录下执行以下命令(随机找个端口号进行激活)

```
curl -G "node03:${(<./executor.port)}/executor?action=activate" && echo
```

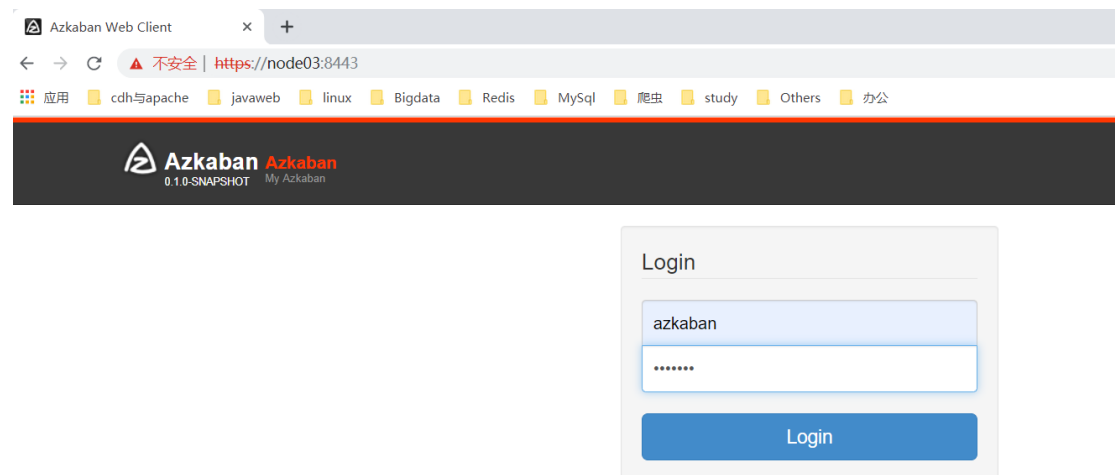
```
[root@node03 azkaban-exec-server-3.51.0]# curl -G "node03:${(<./executor.port)}/executor?action=activate" && echo
{"status":"success"}
[root@node03 azkaban-exec-server-3.51.0]#
```

第三步：启动 azkaban-web-server

```
cd /export/servers/azkaban-web-server-3.51.0/
bin/start-web.sh
```

访问地址：

<https://node03:8443>



修改 linux 的时区问题

由于先前做好了时钟同步，所以不用担心时区问题，不需要修改时区了

注：先配置好服务器节点上的时区

- 1、先生成时区配置文件 Asia/Shanghai，用交互式命令 `tzselect` 即可
- 2、拷贝该时区文件，覆盖系统本地时区配置
`cp /usr/share/zoneinfo/Asia/Shanghai /etc/localtime`

Azkaban 实战

Azkaba 内置的任务类型支持 `command`、`java`

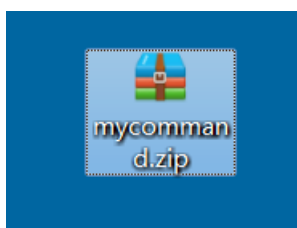
Command 类型单一 job 示例

创建 job 描述文件

创建文本文件，更改名称为 `mycommand.job`
注意后缀.txt 一定不要带上，保存为格式为 UFT-8 without bom
内容如下

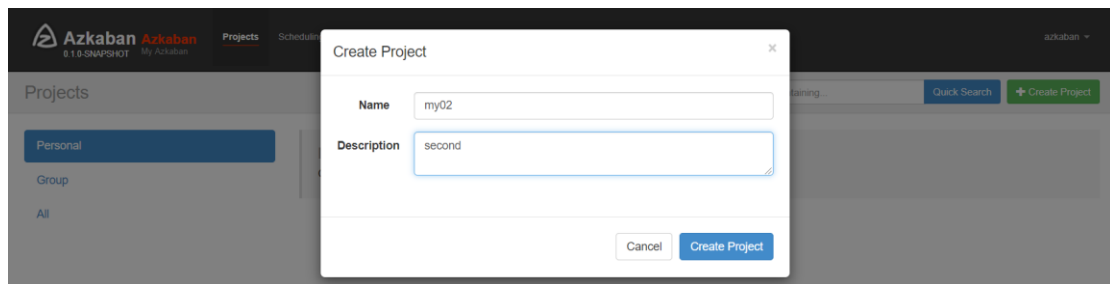
```
type=command  
command=echo 'hello world'
```

将 job 资源文件打包成 zip 文件

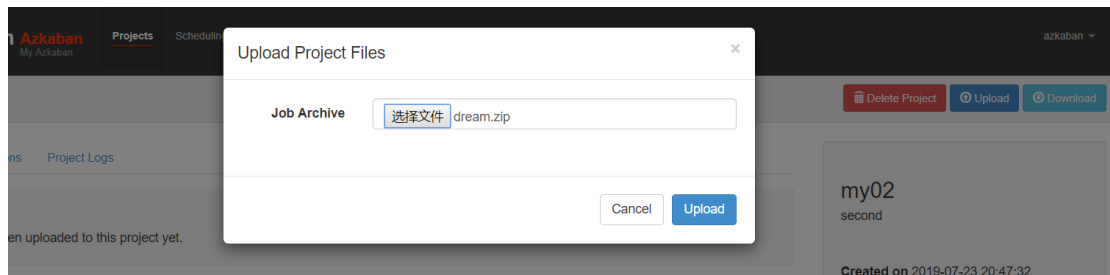


创建 project 并上传压缩包

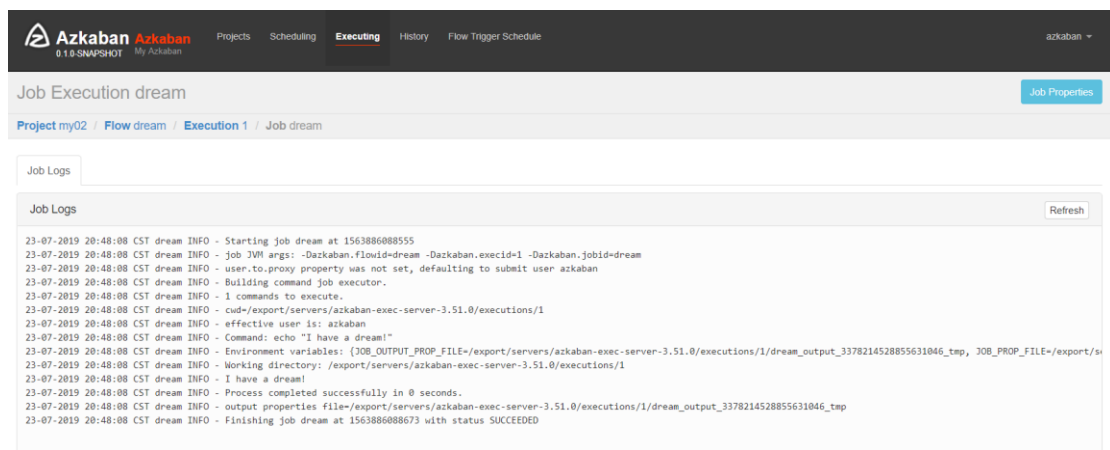
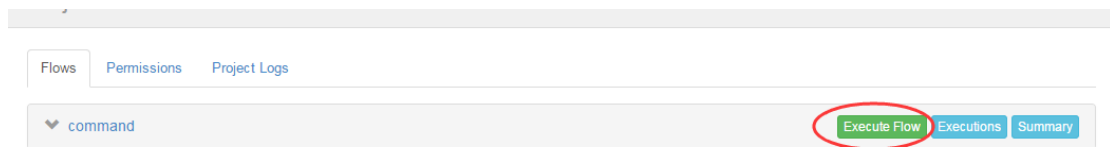
通过 azkaban 的 web 管理平台创建 project 并上传 job 压缩包
首先创建 project



上传 zip 包



启动执行 job



Command 类型多 job 工作流 flow

1、创建有依赖关系的多个 job 描述

第一个 job: foo.job

```
type=command
command=echo 'foo'
```

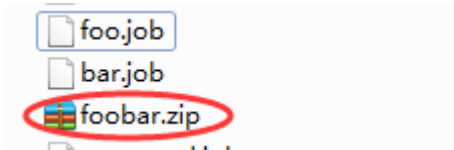
第二个 job: bar.job 依赖 foo.job

```
type=command
```

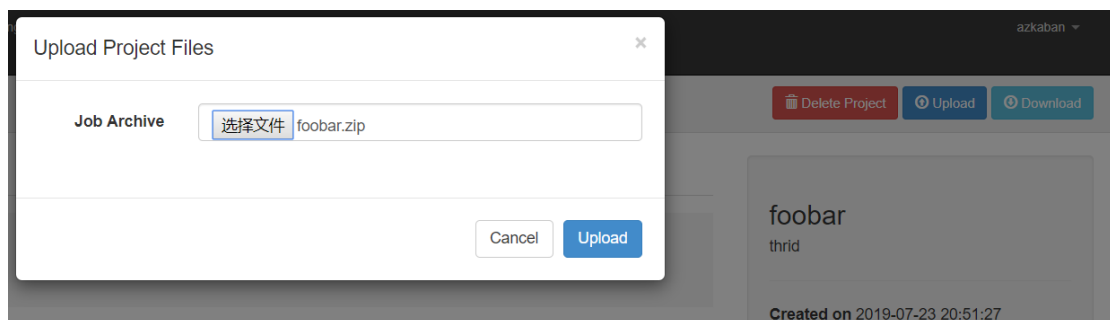
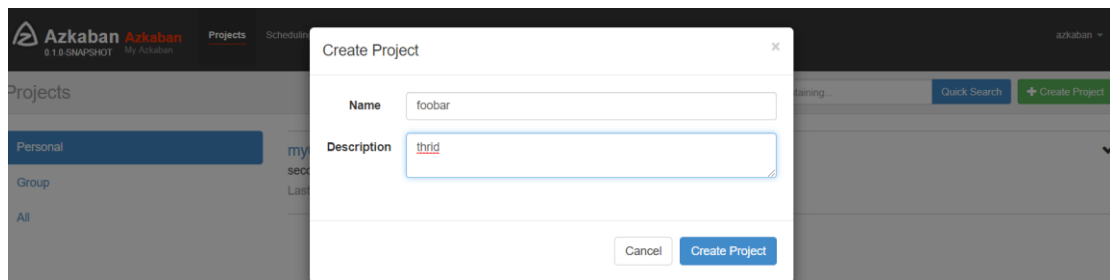


```
dependencies=foo  
command=echo 'bar'
```

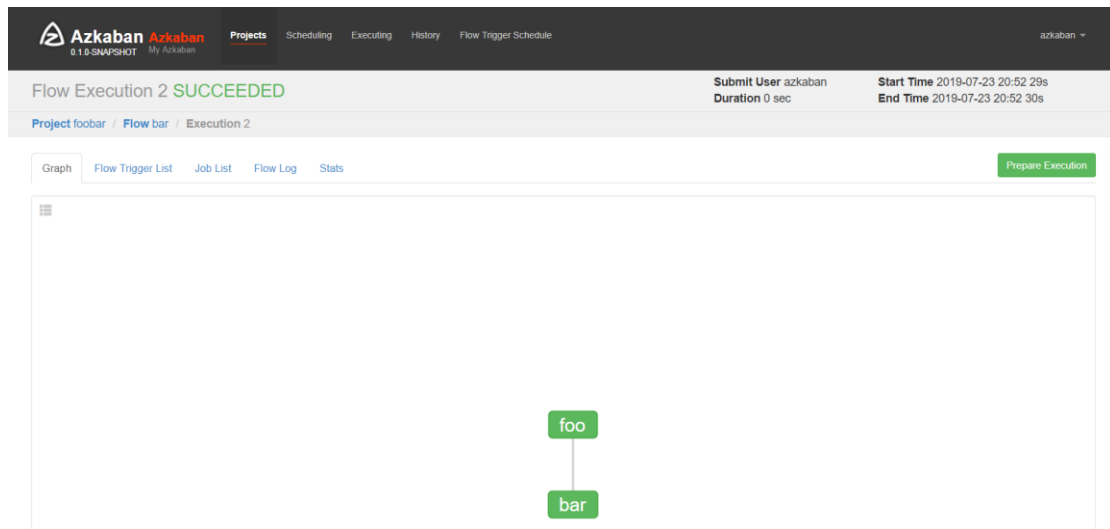
2、将所有 job 资源文件打到一个 zip 包中



3、在 azkaban 的 web 管理界面创建工程并上传 zip 包



4、启动 workflow flow



Flow Execution 2 SUCCEEDED					Submit User azkaban	Start Time 2019-07-23 20:52 29s
					Duration 0 sec	End Time 2019-07-23 20:52 30s
Project foobar / Flow bar / Execution 2						
Graph Flow Trigger List Job List Flow Log Stats					Prepare Execution	
Name	Type	Timeline	Start Time	End Time	Elapsed	Status
foo	command	<div></div>	2019-07-23 20:52 29s	2019-07-23 20:52 29s	0 sec	Success
bar	command	<div></div>	2019-07-23 20:52 29s	2019-07-23 20:52 30s	0 sec	Success

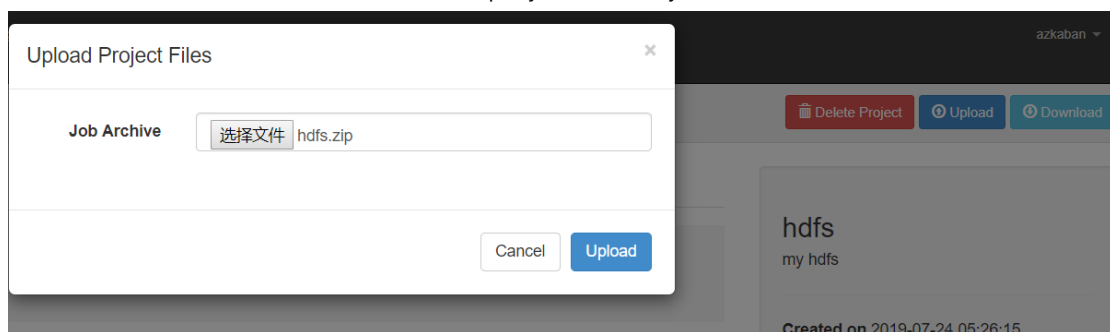
HDFS 操作任务(通过 azkaban 在 hdfs 上创建一个目录)

1、创建 job 描述文件 fs.job

```
type=command
command=/export/servers/hadoop-2.6.0-cdh5.14.0/bin/hdfs dfs -mkdir /azkaban
```

2、将 job 资源文件打包成 zip 文件

3、通过 azkaban 的 web 管理平台创建 project 并上传 job 压缩包



4、启动执行该 job



Flow Execution 3 RUNNING

Submit User azkaban

Duration 3 sec

Start Time 2019-07-24 05:27 41s

End Time -

Project hdfs / Flow hdfs / Execution 3

Graph

Flow Trigger List

Job List

Flow Log

Stats

Kill

Pause

Flow log

Refresh

24-07-2019 05:27:41 CST hdfs INFO - Assigned executor : node03.hadoop.com:38137

24-07-2019 05:27:41 CST hdfs INFO - Running execid:3 flow:hdfs project:3 version:1

24-07-2019 05:27:41 CST hdfs INFO - Updating initial flow directory.

24-07-2019 05:27:41 CST hdfs INFO - Fetching job and shared properties.

24-07-2019 05:27:41 CST hdfs INFO - Starting flow

24-07-2019 05:27:41 CST hdfs INFO - Running flow 'hdfs'.

24-07-2019 05:27:41 CST hdfs INFO - Configuring Azkaban metrics tracking for jobrunner object

24-07-2019 05:27:41 CST hdfs INFO - Submitting job 'hdfs' to run.

24-07-2019 05:27:41 CST hdfs INFO - Created file appender for job hdfs

24-07-2019 05:27:41 CST hdfs INFO - Attached file appender for job hdfs

24-07-2019 05:27:41 CST hdfs INFO - Job Started: hdfs

Job Execution hdfs

Job Properties

Project hdfs / Flow hdfs / Execution 3 / Job hdfs

Job Logs

Refresh

24-07-2019 05:27:41 CST hdfs INFO - Starting job hdfs at 1563917261933

24-07-2019 05:27:41 CST hdfs INFO - job JVM args: -Dazkaban.flowid=hdfs -Dazkaban.execid=3 -Dazkaban.jobid=hdfs

24-07-2019 05:27:41 CST hdfs INFO - user.to.proxy property was not set, defaulting to submit user azkaban

24-07-2019 05:27:41 CST hdfs INFO - Building command job executor.

24-07-2019 05:27:41 CST hdfs INFO - 1 commands to execute.

24-07-2019 05:27:41 CST hdfs INFO - cmd=/export/servers/azkaban-exec-server-3.51.0/executions/3

24-07-2019 05:27:41 CST hdfs INFO - effective user is: azkaban

24-07-2019 05:27:41 CST hdfs INFO - Command: /export/servers/hadoop-2.6.0-cdh5.14.0/bin/hdfs dfs -mkdir /azkaban

24-07-2019 05:27:41 CST hdfs INFO - Environment variables: {JOB_OUTPUT_PROP_FILE=/export/servers/azkaban-exec-server-3.51.0/executions/3/hdfs_output_1437482742516719721_tmp, JOB_PROP_FILE=/export/ser

24-07-2019 05:27:41 CST hdfs INFO - Working directory: /export/servers/azkaban-exec-server-3.51.0/executions/3

24-07-2019 05:27:46 CST hdfs INFO - Process completed successfully in 4 seconds.

24-07-2019 05:27:46 CST hdfs INFO - output properties file=/export/servers/azkaban-exec-server-3.51.0/executions/3/hdfs_output_1437482742516719721_tmp

24-07-2019 05:27:46 CST hdfs INFO - Finishing job hdfs at 1563917266522 with status SUCCEEDED

Hadoop

Overview

Datanodes

Snapshot

Startup Progress

Utilities

Browse Directory

/

Go!

Permission	Owner	Group	Size	Last Modified	Replication	Block Size	Name
drwxr-xr-x	kxb	supergroup	0 B	Thu Jul 18 13:29:06 +0800 2019	0	0 B	a
drwxr-xr-x	kxb	supergroup	0 B	Thu Jul 18 13:15:36 +0800 2019	0	0 B	abc
drwxr-xr-x	root	supergroup	0 B	Mon Jul 22 14:50:58 +0800 2019	0	0 B	avro
drwxr-xr-x	root	supergroup	0 B	Wed Jul 24 05:27:46 +0800 2019	0	0 B	azkaban

MAPREDUCE 任务

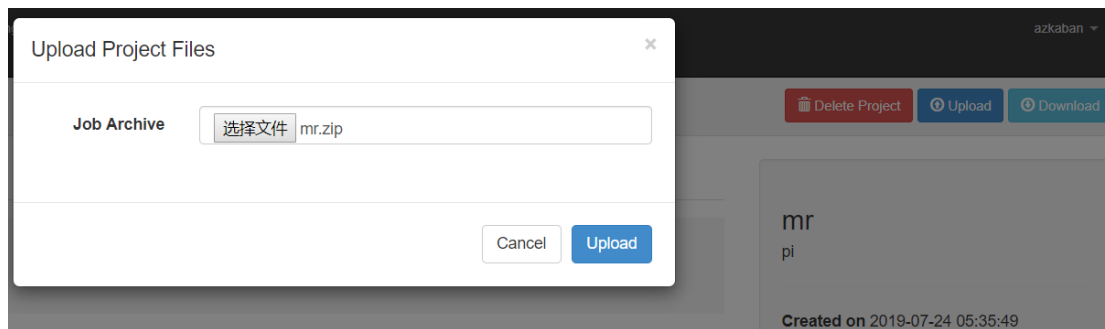
Mr 任务依然可以使用 command 的 job 类型来执行

1、创建 job 描述文件，及 mr 程序 jar 包（示例中直接使用 hadoop 自带的 example jar）

type=command

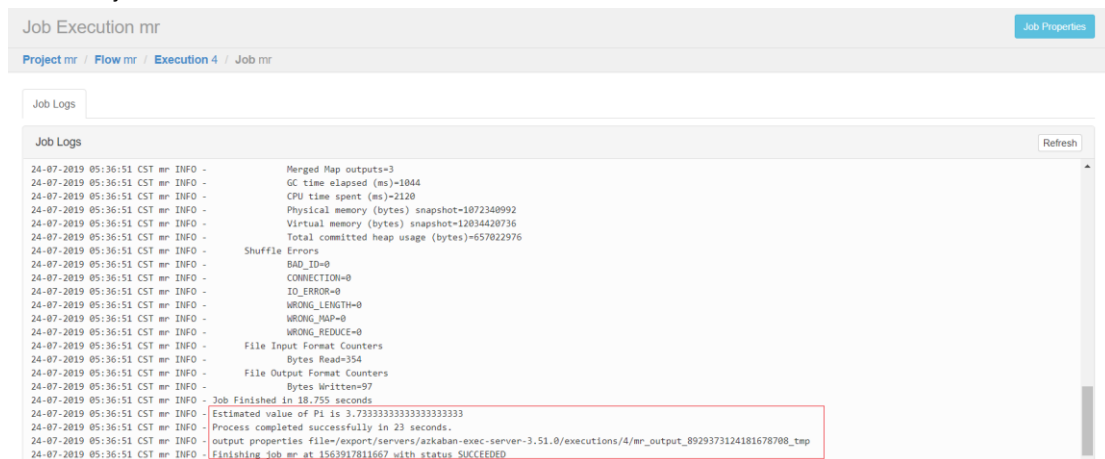
command=/export/servers/hadoop-2.6.0-cdh5.14.0/bin/hadoop jar hadoop-mapreduce-examples-2.6.0-cdh5.14.0.jar pi 3 5

2、将所有 job 资源文件打到一个 zip 包中



3、在 azkaban 的 web 管理界面创建工程并上传 zip 包

4、启动 job



HIVE 脚本任务

- 创建 job 描述文件和 hive 脚本

Hive 脚本: hive.sql

```
create database if not exists azhive;
use azhive;
create table if not exists aztest(id string,name string) row format delimited
fields terminated by '\t';
```

Job 描述文件: hive.job

```
type=command
command=/export/servers/hive-1.1.0-cdh5.14.0/bin/hive -f 'hive.sql'
```

将所有 job 资源文件打到一个 zip 包中

在 azkaban 的 web 管理界面创建工程并上传 zip 包

启动 job

Flow Execution 5 SUCCEEDED

Submit User azkaban

Duration 12 sec

Start Time 2019-07-24 05:44 18s

End Time 2019-07-24 05:44 31s

Project hive / Flow hive / Execution 5

Graph

Flow Trigger List

Job List

Flow Log

Stats

Prepare Execution

Name	Type	Timeline	Start Time	End Time	Elapsed	Status	Details
hive	command		2019-07-24 05:44 18s	2019-07-24 05:44 31s	12 sec	Success	Details

```
cd /export/servers/hive-1.1.0-cdh5.14.0/
bin/hive
show databases;
```

```

[root@node03 hive-1.1.0-cdh5.14.0]# bin/hive
which: no hbase in (:/export/servers/hadoop-2.6.0-cdh5.14.0/bin:/export/se
/servers/jdk1.8.0_141/bin:/usr/lib64/qt-3.3/bin:/usr/local/sbin:/usr/local
in)

Logging initialized using configuration in jar:file:/export/servers/hive-1
14.0.jar!/hive-log4j.properties
WARNING: Hive CLI is deprecated and migration to Beeline is recommended.
hive (default)> show databases;
OK
database_name
azhive
default
myhive010
Time taken: 5.053 seconds, Fetched: 3 row(s)
hive (default)>
```

```
use azhive;
show tables;
```

```

hive (default)> use azhive;
OK
Time taken: 0.047 seconds
hive (azhive)> show tables;
OK
tab_name
aztest
Time taken: 0.047 seconds, Fetched: 1 row(s)
hive (azhive)>
```

azkaban 的定时任务

使用 azkaban 的 scheduler 功能可以实现对我们的作业任务进行定时调度功能

Flow View

Right click on the jobs to disable and enable jobs in the flow.

Notification

Failure Options

Concurrent

Flow Parameters

dream

Schedule

Cancel

Execute

Schedule Flow Options

✕

All schedules are basead on the server timezone: **Asia/Shanghai**.

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.](#)

定时任务的表达式写法

0 * * ? *

Reset

Next 10 scheduled executions:

Cancel

Schedule

Baidu

百度

crontab

百度一下

网页

资讯

视频

图片

知道

文库

贴吧

采购

地图

更多»

百度为您找到相关结果约11,000,000个

搜索工具

您可以仅查看：[英文结果](#)

[Linux下的crontab定时执行任务命令详解 - 回家的流浪者 - 博客园](#)

2019年4月1日 - 1、如果两个文件都不存在,则只有root用户才能使用crontab命令。2、如果cron.allow存在但cron.deny不存在,则只有列在cron.allow文件里的用户才能使用c...
<https://www.cnblogs.com/longjs...> - 百度快照

[crontab执行时间计算 - 在线工具](#)

类型: Linux Java(Spring) crontab表达式: 执行时间 0 */12 * * * [user] [command] 请只输入红色部分。接下来7次的执行时间:....
tool.lu/crontab/ - 百度快照

[crontab的使用方法介绍 - python小白的逆袭之路 - CSDN博客](#)

2018年6月9日 - 使用crontab你可以在指定的时间执行一个shell脚本或者一系列Linux命令。例如系统管理员安排一个备份任务使其每天都运行...
[CSDN](#) - 百度快照

[在线Cron表达式生成器](#)

通过这个生成器,您可以在线生成任务调度比如Quartz的Cron表达式,对Quartz Cron 表达式的可视化双向解析和生成。
cron.qqe2.com/ - 百度快照

指定每一天的 2 点 01 分开始执行

秒分钟小时日月周年

分钟 允许的通配符[, * /]

周期从

1

-

2

分钟

从

0

分钟开始,每

1

分钟执行一次

指定

☐ 00

☒ 01

☐ 02

☐ 03

☐ 04

☐ 05

☐ 06

☐ 07

☐ 08

☐ 09

☐ 10

☐ 11

☐ 12

☐ 13

☐ 14

☐ 15

☐ 16

☐ 17

☐ 18

☐ 19

☐ 20

☐ 21

☐ 22

☐ 23

☐ 24

☐ 25

☐ 26

☐ 27

☐ 28

☐ 29

☐ 30

☐ 31

☐ 32

☐ 33

☐ 34

☐ 35

☐ 36

☐ 37

☐ 38

☐ 39

☐ 40

☐ 41

☐ 42

☐ 43

☐ 44

☐ 45

☐ 46

☐ 47

☐ 48

☐ 49

☐ 50

☐ 51

☐ 52

☐ 53

☐ 54

☐ 55

☐ 56

☐ 57

☐ 58

☐ 59

表达式

秒分钟小时日月星期年

表达式字段: 012*?**

Cron 表达式: 0 1 2 * * *

反解析到UI

秒
分钟
小时
日
月
周
年

☐ 小时 允许的通配符[, - * /]

☐ 周期从 0 - 2 小时

☐ 从 0 小时开始,每 1 小时执行一次

☒ 指定

AM: ☐ 00 ☐ 01 ☒ 02 ☐ 03 ☐ 04 ☐ 05 ☐ 06 ☐ 07 ☐ 08 ☐ 09 ☐ 10 ☐ 11
PM: ☐ 12 ☐ 13 ☐ 14 ☐ 15 ☐ 16 ☐ 17 ☐ 18 ☐ 19 ☐ 20 ☐ 21 ☐ 22 ☐ 23

表达式

秒
分钟
小时
日
月
星期
年

表达式字段: 0 1 2 * * ? *

Cron 表达式: 0 1 2 * * ? * 反解析到UI

最近5次运行时间:

2019/7/25 2:01:00
2019/7/26 2:01:00
2019/7/27 2:01:00
2019/7/28 2:01:00
2019/7/29 2:01:00

此表达式从秒开始:

表达式

秒
分钟
小时
日
月
星期
年

表达式字段: 0 1 2 * * ? *

Cron 表达式: 0 1 2 * * ? * 反解析到UI

最近5次运行时间:

Azkaban
Azkaban
6.1.0-SNAPSHOT
My Azkaban

Projects
Scheduling
Executing
History
Flow Trigger Schedule

azkaban

Scheduled Flows

* Click column headers to sort.

#	ID	Flow	Project	Submitted By	First Scheduled to Run	Next Execution Time	Repeats Every	Cron Expression	Execution Options	Has SLA	Action
1	1	dream	my02	azkaban	2019-07-24 06:02:15	2019-07-25 02:01:00	Not Applicable	0 1 2 ? * *	Show	false	Remove Schedule Set SLA

Schedule Flow Options



All schedules are based on the server timezone: Asia/Shanghai.

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	<input type="text" value="1"/>
Hours	<input type="text" value="2"/>
Day of Month	<input type="text" value="?"/>
Month	<input type="text" value="*/"/>
Day of Week	<input type="text" value="*/"/>
Year	<input type="text"/>

Special Characters:

- * any value
- , value list separators
- range of values
- / step values
- 0-23 allowed values

[Detailed instructions.](#)

Reset

Next 10 scheduled executions for this cron expression only:

- 2019-07-25T02:01:00
- 2019-07-26T02:01:00
- 2019-07-27T02:01:00
- 2019-07-28T02:01:00
- 2019-07-29T02:01:00