大数据Hadoop高薪直通车课程

工作流调度框架Oozie

讲师：轩宇（北风网版权所有)

课程大纲

1. Hadoop 调度框架
2. Oozie 功能架构
3. Oozie 安装部署
4. Oozie 工作流调度
5. Oozie 协作调度

Map-Reduce Action

A map-reduce action can be configured to perform file system cleanup and directory creation before starting the map reduce job.

The workflow job will wait until the Hadoop map/reduce job completes before continuing to the next action in the workflow execution path.

The counters of the Hadoop job and job exit status (=FAILED=, KILLED or SUCCEEDED) must be available to the workflow job after the Hadoop jobs ends.

The map-reduce action has to be configured with all the necessary Hadoop JobConf properties to run the Hadoop map/reduce job.

```
<workflow-app name="foo-wf" xmlns="uri:oozie:workflow:0.1">
    ...
    <action name="myfirstHadoopJob">
        <map-reduce>
            <job-tracker>foo:8021</job-tracker>
            <name-node>bar:8020</name-node>
            <prepare>
                <delete path="hdfs://foo:8020/usr/tucu/output-data"/>
            </prepare>
            <job-xml>/myfirstjob.xml</job-xml>
            <configuration>
                <property>
                    <name>mapred.input.dir</name>
                    <value>/usr/tucu/input-data</value>
                </property>
                <property>
                    <name>mapred.output.dir</name>
                    <value>/usr/tucu/input-data</value>
                </property>
            </configuration>
        </map-reduce>
        <ok to="myNextAction"/>
        <error to="errorCleanup"/>
    </action>
    ...
</workflow-app>
```

Action Nodes

- Hive Action

http://archive.cloudera.com/cdh5/cdh/5/oozie-4.0.0-cdh5.3.6/DG_HiveActionExtension.html

- Sqoop Action

http://archive.cloudera.com/cdh5/cdh/5/oozie-4.0.0-cdh5.3.6/DG_SqoopActionExtension.html

- Shell Action

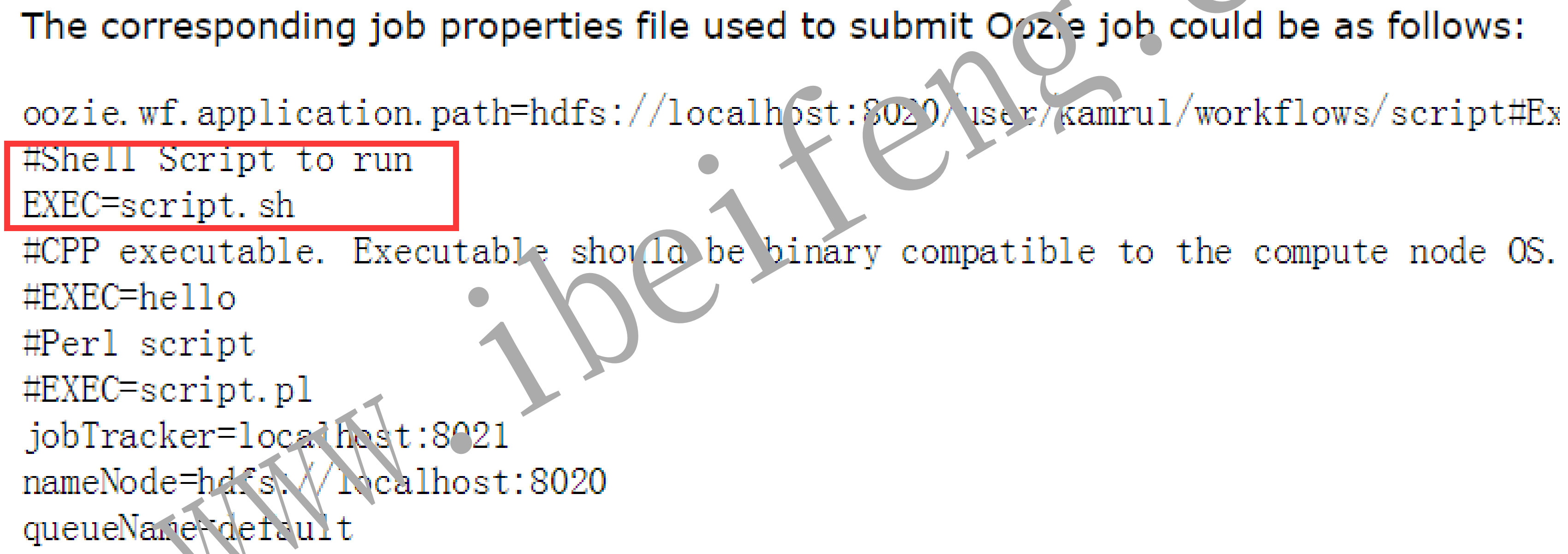
http://archive.cloudera.com/cdh5/cdh/5/oozie-4.0.0-cdh5.3.6/DG_ShellActionExtension.html

Shell Action

How to run any shell script or perl script or CPP executable

```
<workflow-app xmlns='uri:oozie:workflow:0.3' name='shell-wf'>
    <start to='shell1' />
    <action name='shell1'>
        <shell xmlns="uri:oozie:shell-action:0.1">
            <job-tracker>${jobTracker}</job-tracker>
            <name-node>${nameNode}</name-node>
            <configuration>
                <property>
                  <name>mapred.job.queue.name</name>
                  <value>${queueName}</value>
                </property>
            </configuration>
            <exec>${EXEC}</exec>
            <argument>A</argument>
            <argument>B</argument>
            <file>${EXEC}#${EXEC}</file> <!--Copy the executable to compute node's current working directory --
        </shell>
        <ok to="end" />
        <error to="fail" />
    </action>
    <kill name="fail">
        <message>Script failed, error message[${wf:errorMessage(wf:lastErrorNode())}]</message>
    </kill>
    <end name='end' />
</workflow-app>
```

Shell Action



The corresponding job properties file used to submit Oozie job could be as follows:

```
oozie.wf.application.path=hdfs://localhost:8020/user/kamrul/workflows/script#Ex
#Shell Script to run
EXEC=script.sh
#CPP executable. Executable should be binary compatible to the compute node OS.
#EXEC=hello
#Perl script
#EXEC=script.pl
jobTracker=localhost:8021
nameNode=hdfs://localhost:8020
queueName=default
```

课程大纲

1 Hadoop 调度框架

2 Oozie 功能架构

3 Oozie 安装部署

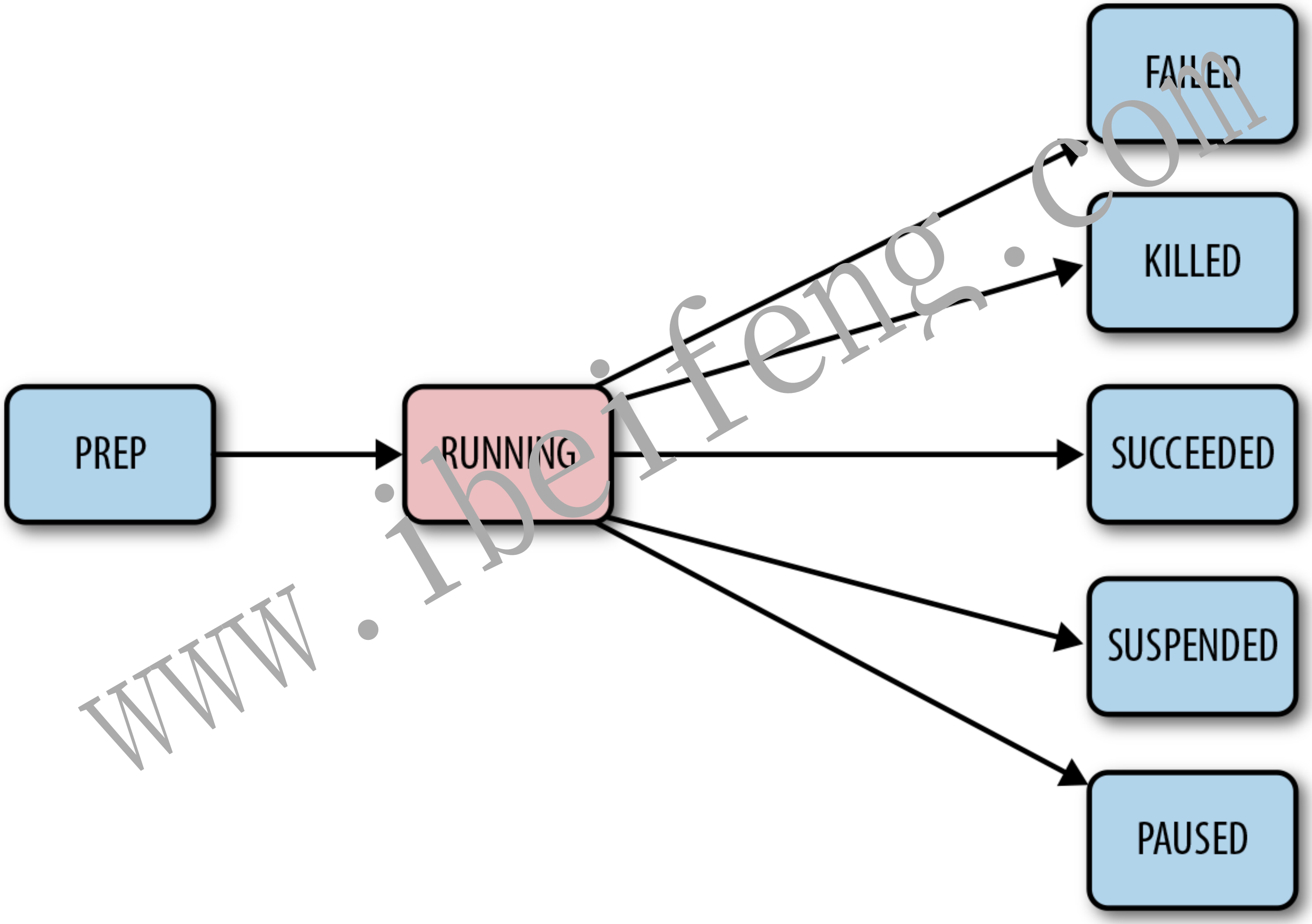
4 Oozie 工作流调度

5 Oozie 协作调度

Oozie Coordinator

```
<coordinator-app name="cron-coord" frequency="0/10 1/2 * * *" start="${start}" end="${end}" timezone="UTC"
                xmlns="uri:oozie:coordinator:0.2">
        <action>
        <workflow>
            <app-path>${workflowAppUri}</app-path>
            <configuration>
                <property>
                    <name>jobTracker</name>
                    <value>${jobTracker}</value>
                </property>
                <property>
                    <name>nameNode</name>
                    <value>${nameNode}</value>
                </property>
                <property>
                    <name>queueName</name>
                    <value>${queueName}</value>
                </property>
            </configuration>
        </workflow>
    </action>
</coordinator-app>
```

Coordinator job lifecycle

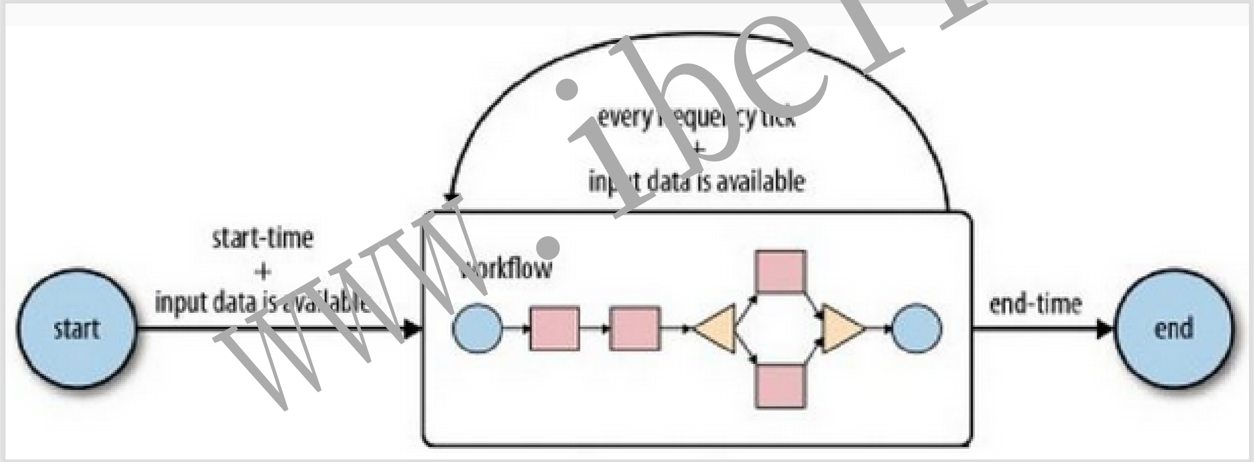


Oozie

Oozie 是用于管理 Hadoop Job 的工作流调度平台，工作流创建好之后，就需要 `Coordinator` 来进行调度了。Coordinator 进行调度依赖时间条件和数据是否可用。Coordinator 中几个关键的概念：

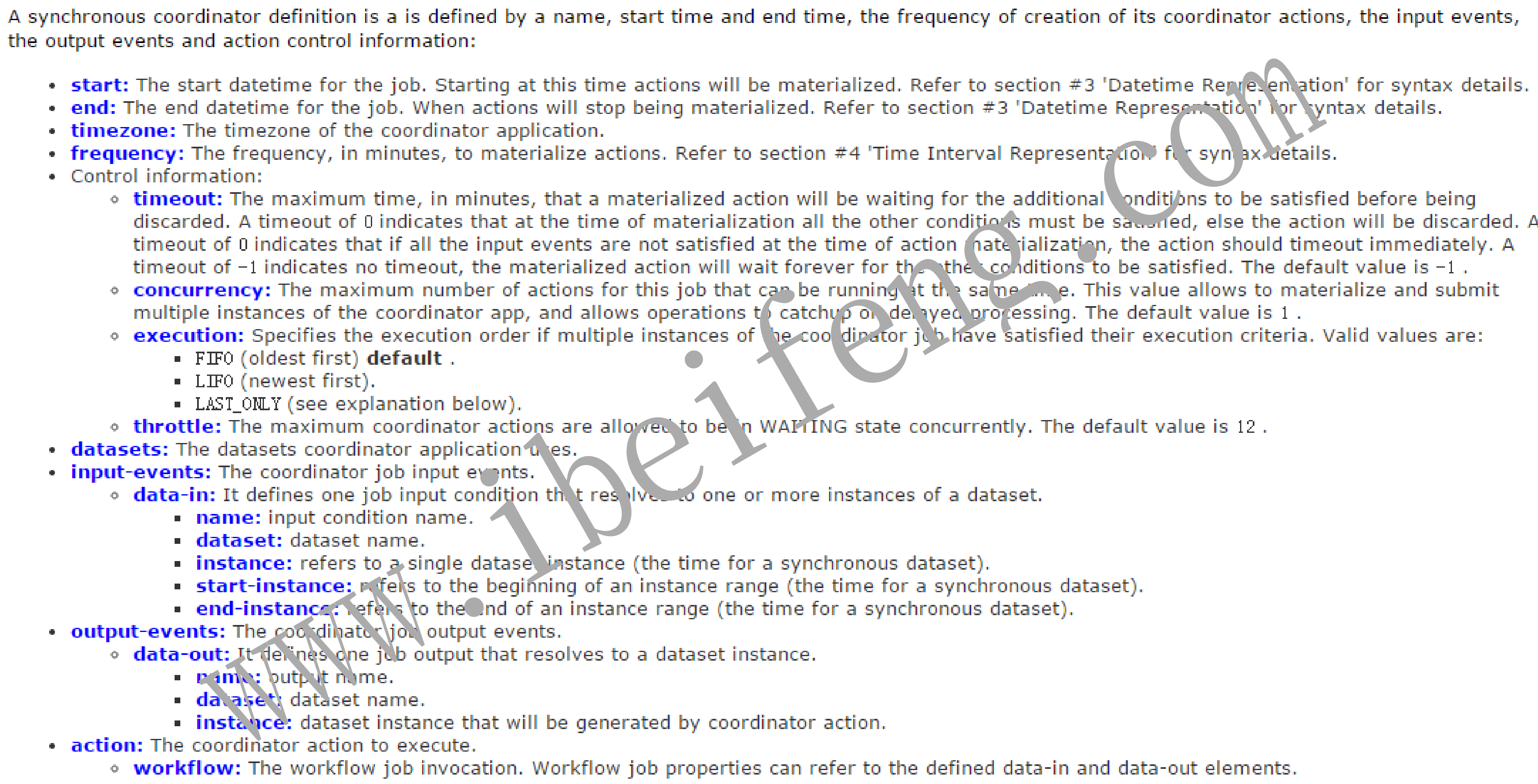
- ***Coordinator Action***，即是一个 Workflow Job ，Job 在满足一系列条件后开始执行；
- ***Coordinator Application***，定义了 Coordinator Action 的执行条件，包括 Coordinator Action 执行的时间频率，以及 Coordinator Action 需要被执行的开始时间和不再被执行的结束时间；
- ***Coordinator Job***，即一个 Coordinator Application 的运行实例。

一个 Coordinator Application 的定义包括开始时间、结束时间、执行频率、输入数据、输出数据和被执行的工作流。一个 Coordinator Application 在开始时间起周期性的执行工作流，如下图所示。



开始执行时，Coordinator Job 首先会检查是否输入数据能够获取到。当输入数据存在可用时，Workflow Job 处理输入数据，并产生对应的输出数据。如果输入数据不存在不可用时，Workflow Job 的执行过程推迟，直到输入数据可用。若结束时间条件满足时，Coordinator Job 不再执行。

Synchronous Coordinator Application Definition



A synchronous coordinator definition is a is defined by a name, start time and end time, the frequency of creation of its coordinator actions, the input events, the output events and action control information:

- **start:** The start datetime for the job. Starting at this time actions will be materialized. Refer to section #3 'Datetime Representation' for syntax details.
- **end:** The end datetime for the job. When actions will stop being materialized. Refer to section #3 'Datetime Representation' for syntax details.
- **timezone:** The timezone of the coordinator application.
- **frequency:** The frequency, in minutes, to materialize actions. Refer to section #4 'Time Interval Representation' for syntax details.
- Control information:
  - **timeout:** The maximum time, in minutes, that a materialized action will be waiting for the additional conditions to be satisfied before being discarded. A timeout of `0` indicates that at the time of materialization all the other conditions must be satisfied, else the action will be discarded. A timeout of `0` indicates that if all the input events are not satisfied at the time of action materialization, the action should timeout immediately. A timeout of `-1` indicates no timeout, the materialized action will wait forever for the other conditions to be satisfied. The default value is `-1` .
  - **concurrency:** The maximum number of actions for this job that can be running at the same time. This value allows to materialize and submit multiple instances of the coordinator app, and allows operations to catchup on delayed processing. The default value is `1` .
  - **execution:** Specifies the execution order if multiple instances of the coordinator job have satisfied their execution criteria. Valid values are:
    - `FIFO` (oldest first) **default** .
    - `LIFO` (newest first).
    - `LAST_ONLY` (see explanation below).
  - **throttle:** The maximum coordinator actions are allowed to be in WAITING state concurrently. The default value is `12` .
- **datasets:** The datasets coordinator application uses.
- **input-events:** The coordinator job input events.
  - **data-in:** It defines one job input condition that resolves to one or more instances of a dataset.
    - **name:** input condition name.
    - **dataset:** dataset name.
    - **instance:** refers to a single dataset instance (the time for a synchronous dataset).
    - **start-instance:** refers to the beginning of an instance range (the time for a synchronous dataset).
    - **end-instance:** refers to the end of an instance range (the time for a synchronous dataset).
- **output-events:** The coordinator job output events.
  - **data-out:** It defines one job output that resolves to a dataset instance.
    - **name:** output name.
    - **dataset:** dataset name.
    - **instance:** dataset instance that will be generated by coordinator action.
- **action:** The coordinator action to execute.
  - **workflow:** The workflow job invocation. Workflow job properties can refer to the defined data-in and data-out elements.

http://archive.cloudera.com/cdh5/cdh/5/oozie-4.0.0-cdh5.3.6/CoordinatorFunctionalSpec.html#a6.2._Synchronous_Coordinator_Application_Example

Control information

```
<controls>
     <timeout>[TIME_PERIOD]</timeout>
     <concurrency>[CONCURRENCY]</concurrency>
     <execution>[EXECUTION_STRATEGY]</execution>
</controls>
```

control元素定义了一个Coordinator Job的控制信息，主要包括如下三个配置元素：

元素名称	含义说明
timeout	超时时间，单位为分钟。当一个Coordinator Job启动的时候，会初始化多个Coordinator动作，timeout用来限制这个初始化过程。默认值为-1，表示永远不超时，如果为0 则总是超时。
concurrency	并发数，指多个Coordinator Job并发执行，默认值为1。
execution	配置多个Coordinator Job并发执行的策略：默认是FIFO。另外还有两种：LIFO（最新的先执行）、LAST_ONLY（只执行最新的Coordinator Job，其它的全部丢弃）。
throttle	一个Coordinator Job初始化时，允许Coordinator动作处于WAITING状态的最大数量。

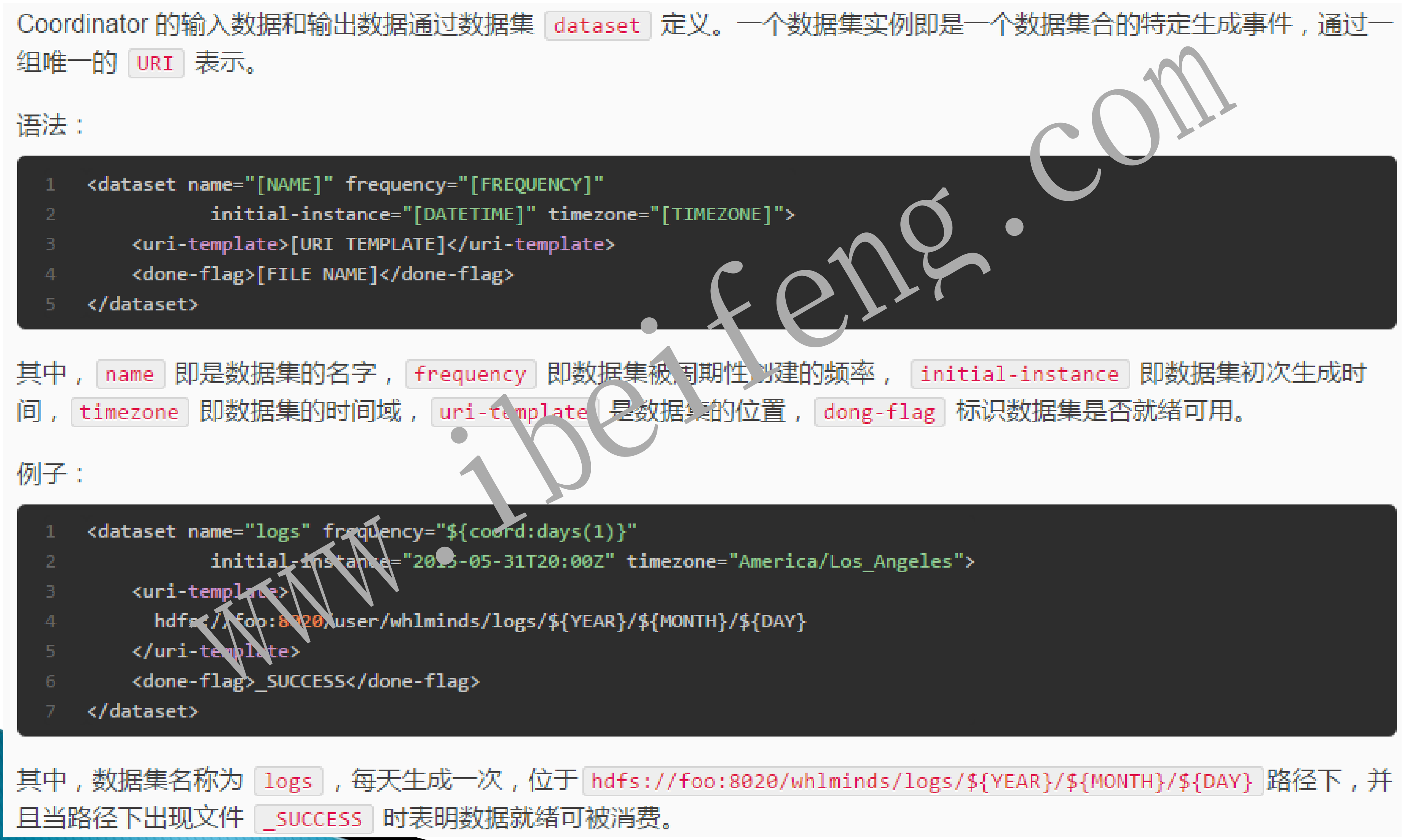
Coordinator Job中有一个Dataset的概念，它可以为实际计算提供计算的数据，主要是指HDFS上的数据目录或文件，能够配置数据集生成的频率（Frequency）、URI模板、时间等信息，下面看一下dataset的语法格式：

```
<dataset name="[NAME]" frequency="[FREQUENCY]" initial-instance="[DATETIME]" timezone="[TIMEZONE]">
     <uri-template>[URI TEMPLATE]</uri-template>
     <done-flag>[FILE NAME]</done-flag>
</dataset>
```

举例如下：

```
<dataset name="stats_hive_table" frequency="${coord:days(1)}" initial-instance="2014-03-05T00:00Z" timezone="America/Los_Angeles">
     <uri-template>
          hdfs://m1:9000/hive/warehouse/user_events/${YEAR}${MONTH}/${DAY}/data
     </uri-template>
     <done-flag>donefile.flag</done-flag>
</dataset>
```

上面会每天都会生成一个用户事件表，可以供Hive查询分析，这里指定了这个数据集的位置，后续计算会使用这部分数据。其中，uri-template指定了一个匹配的模板，满足这个模板的路径都会被作为计算的基础数据。



Coordinator 的输入数据和输出数据通过数据集 `dataset` 定义。一个数据集实例即是一个数据集合的特定生成事件，通过一组唯一的 `URI` 表示。

语法：

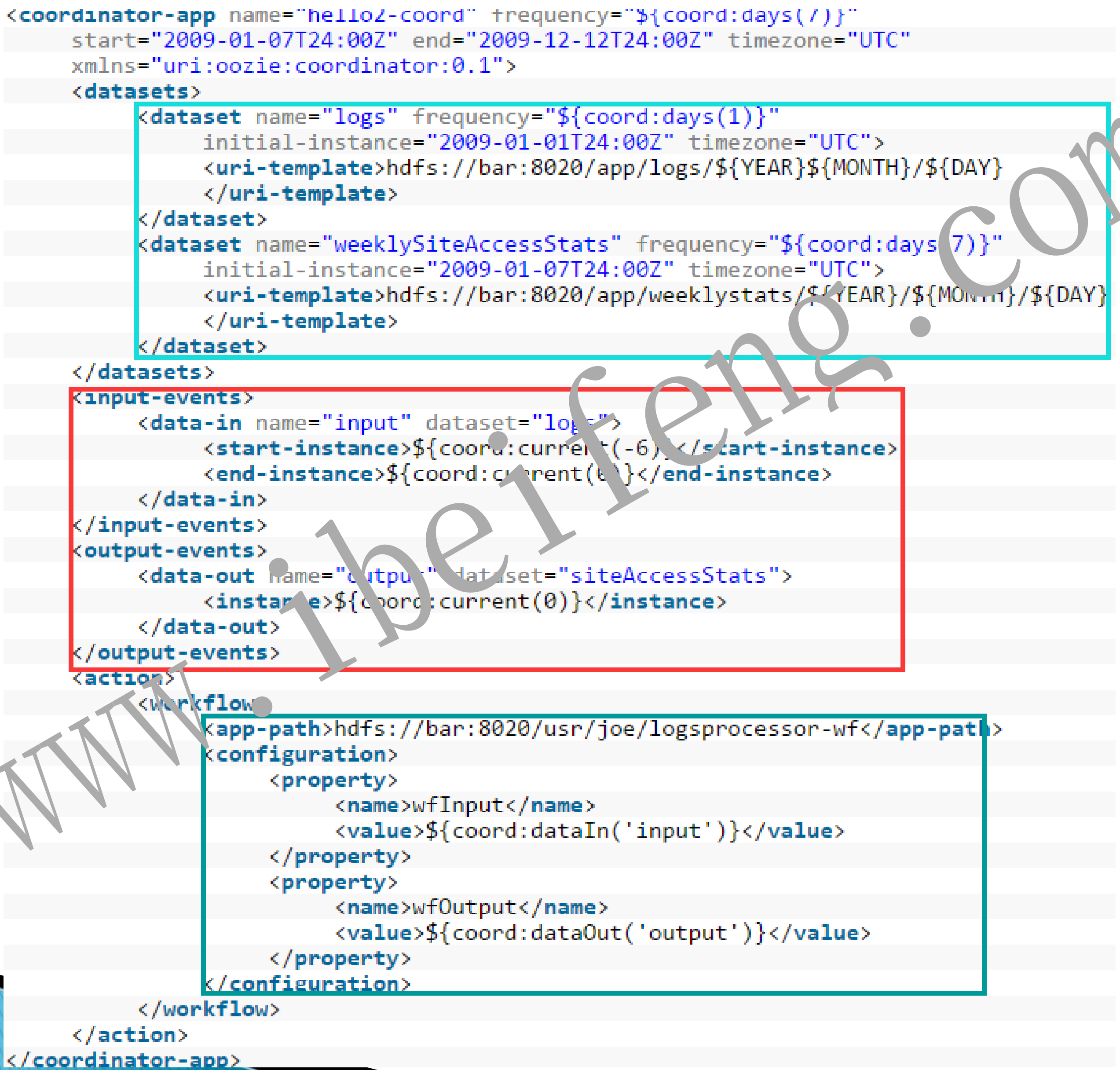
```
<dataset name="[NAME]" frequency="[FREQUENCY]"
           initial-instance="[DATETIME]" timezone="[TIMEZONE]">
    <uri-template>[URI TEMPLATE]</uri-template>
    <done-flag>[FILE NAME]</done-flag>
</dataset>
```

其中，`name` 即是数据集的名字，`frequency` 即数据集被周期性创建的频率，`initial-instance` 即数据集初次生成时间，`timezone` 即数据集的时间域，`uri-template` 是数据集的位置，`dong-flag` 标识数据集是否就绪可用。

例子：

```
<dataset name="logs" frequency="${coord:days(1)}"
           initial-instance="2015-05-31T20:00Z" timezone="America/Los_Angeles">
    <uri-template>
      hdfs://foo:8020/user/whlminds/logs/${YEAR}/${MONTH}/${DAY}
    </uri-template>
    <done-flag>_SUCCESS</done-flag>
</dataset>
```

其中，数据集名称为 `logs`，每天生成一次，位于 `hdfs://foo:8020/whlminds/logs/${YEAR}/${MONTH}/${DAY}` 路径下，并且当路径下出现文件 `_SUCCESS` 时表明数据就绪可被消费。



```
<coordinator-app name="hello2-coord" frequency="${coord:days(7)}"
    start="2009-01-07T24:00Z" end="2009-12-12T24:00Z" timezone="UTC"
    xmlns="uri:oozie:coordinator:0.1">
    <datasets>
        <dataset name="logs" frequency="${coord:days(1)}"
            initial-instance="2009-01-01T24:00Z" timezone="UTC">
            <uri-template>hdfs://bar:8020/app/logs/${YEAR}${MONTH}/${DAY}
            </uri-template>
        </dataset>
        <dataset name="weeklySiteAccessStats" frequency="${coord:days(7)}"
            initial-instance="2009-01-07T24:00Z" timezone="UTC">
            <uri-template>hdfs://bar:8020/app/weeklystats/${YEAR}/${MONTH}/${DAY}
            </uri-template>
        </dataset>
    </datasets>
    <input-events>
        <data-in name="input" dataset="logs">
            <start-instance>${coord:current(-6)}</start-instance>
            <end-instance>${coord:current(0)}</end-instance>
        </data-in>
    </input-events>
    <output-events>
        <data-out name="output" dataset="siteAccessStats">
            <instance>${coord:current(0)}</instance>
        </data-out>
    </output-events>
    <action>
        <workflow>
            <app-path>hdfs://bar:8020/usr/joe/logsprocessor-wf</app-path>
            <configuration>
                <property>
                    <name>wfInput</name>
                    <value>${coord:dataIn('input')}</value>
                </property>
                <property>
                    <name>wfOutput</name>
                    <value>${coord:dataOut('output')}</value>
                </property>
            </configuration>
        </workflow>
    </action>
</coordinator-app>
```