Templeton

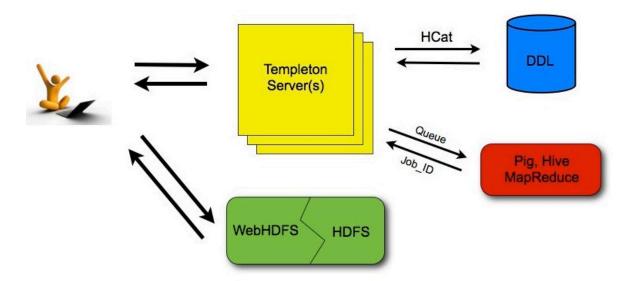## Table of contents

1 Templeton

1.1 Templeton

1.1.1 Introduction

Templeton provides a REST-like web API for HCatalog and related Hadoop components. As shown in the figure below, developers make HTTP requests to access Hadoop MapReduce, Pig, Hive, and HCatalog DDL from within applications. Data and code used by Templeton is maintained in HDFS. HCatalog DDL commands are executed directly when requested. MapReduce, Pig, and Hive jobs are placed in queue by Templeton and can be monitored for progress or stopped as required. Developers specify a location in HDFS into which Templeton should place Pig, Hive, and MapReduce results.

1.1.2 URL format

Templeton resources are accessed using the following URL format:

```
http://yourserver/templeton/v1/resource.format
```

where "*yourserver*" is replaced with your server name, and "*resource.format*" is replaced by the Templeton resource name and output response format. Note that in Templeton version 0.1.0, JSON is the only supported response format.

For example, to check if the Templeton server is running you could access the following URL:

```
http://www.myserver.com/templeton/v1/status.json
```

1.1.3 Security

The current version of Templeton supports two types of security:

* Default security (without additional authentication)
* Authentication via Kerberos

1.1.3.1 Standard Parameters

Every Templeton resource can accept the following parameters to aid in authentication:

* user.name: The user name as a string. Only valid when using default security.
* SPNEGO credentials: When running with Kerberos authentication.

1.1.4 WebHDFS and Code Push

Data and code that are used by Templeton resources must first be placed in Hadoop. The current version of Templeton does not attempt to integrate or replace existing web interfaces that can perform this task, like WebHDFS. (Integration of these functions in some way, perhaps forwarding, is planned for a future release.) When placing files into HDFS is required you can use whatever method is most convienient for you.

1.1.5 Error Codes and Responses

The Templeton server returns the following HTTP status codes.

* **200 OK:** Success!
* **400 Bad Request:** The request was invalid.
* **401 Unauthorized:** Credentials were missing or incorrect.
* **404 Not Found:** The URI requested is invalid or the resource requested does not exist.
* **500 Internal Server Error:** We received an unexpected result.
* **503 Busy, please retry:** The server is busy.

Other data returned directly by Templeton is currently returned in JSON format. JSON responses are limited to 1MB in size. Responses over this limit must be stored into HDFS using provided options instead of being directly returned. If an HCatalog DDL command might return results greater than 1MB, it's suggested that a corresponding Hive request be executed instead.

1.1.6 Project Name

The Templeton project is named after the a character in the award-winning children's novel Charlotte's Web, by E. B. White. The novel's protagonist is a pig named Wilber. Templeton is a rat who helps Wilber by running errands and making deliveries.

1.2 Installation

1.2.1 Introduction

Templeton is deep in the middle of development and does not yet have a smooth install procedure. It is also designed to connect together services that are not normally connected and therefore has a complex configuration. As such, this version of Templeton should only be installed by expert developers.

1.2.2 Requirements

- Ant, version 1.8 or higher
- Tomcat, version 7.0 or higher
- Hadoop, version 0.20.205.0
- ZooKeeper. (Be sure to review and update the ZooKeeper related Templeton configuration.)
- HCatalog. Required to use the ddl resource. (Be sure to review and update the related Templeton configuration.)
- Hadoop Distributed Cache. To use the Hive, Pig, or hadoop/streaming resources, see instructions below for placing the required files in the Hadoop Distributed Cache.

1.2.3 Procedure

1. Ensure that the required related installations are in place.
2. Set the TEMPLETON_HOME environment variable to the base of the Templeton installation. This is used to find the Templeton configuration.
3. Review the Templeton configuration and update or create templeton-site.xml as required. Ensure that site specific component installation locations are accurate, especially the Hadoop configuration path. Configuration variables that use a filesystem path try to have reasonable defaults, but it's always safe to specify a full and complete path.
4. Start Tomcat. (Note: The Tomcat server needs to be restarted after any configuration change.)
5. Build the Templeton war file using ant war from $TEMPLETON_HOME. Copy the Templeton war file to the Tomcat webapps directory.
6. Check that your local install works. Assuming that Tomcat is running on port 8080, the following command would give output similar to that shown.

```
% curl -i http://localhost:8080/templeton/v1/status.json
HTTP/1.1 200 OK
Server: Apache-Coyote/1.1
Content-Type: application/json
Transfer-Encoding: chunked
Date: Tue, 18 Oct 2011 19:35:17 GMT
```

```
{"status": "ok", "version": "v1"}
%
```

1.2.4 Hadoop Distributed Cache

Templeton requires some files be accessible on the Hadoop distributed cache. For example, to avoid the installation of Pig and Hive everywhere on the cluster, Templeton gathers a version of Pig or Hive from the Hadoop distributed cache whenever those resources are invoked. To install the required components in the locations specificed by the default configuration, follow these steps:

1. Download the HCatalog tar.gz file and place it in the distributed cache. (If you need a version that is not yet released, you may need to build it yourself following the HCatalog instructions.)

```
hadoop fs -put /tmp/hcatalog-0.3.0.tar.gz /user/templeton/hcatalog-0.3.0.tar.gz
```

2. Download the Pig tar.gz file and place it into the cache. For example:

```
hadoop fs -put /tmp/pig-0.9.2.tar.gz /user/templeton/pig-0.9.2.tar.gz
```

3. Place `hadoop-streaming.jar` into the cache. For example, use the following command, substituting your path to the jar for the one below.

```
hadoop fs -put $HADOOP_PREFIX/hadoop-0.20.205.0/contrib/streaming/hadoop-
streaming-0.20.205.0.jar \
  /user/templeton/hadoop-streaming.jar
```

The location of these files in the cache, and the location of the installations inside the archives, can be specified using the following Templeton configuration variables. (See the Configuration documentation for more information on changing Templeton configuration parameters.)

Name	Default	Description
templeton.pig.archive	`hdfs:///user/templeton/` `pig-0.9.2.tar.gz`	The path to the Pig archive.
templeton.pig.path	`pig-0.9.2.tar.gz/` `pig-0.9.2/bin/pig`	The path to the Pig executable.
templeton.hive.archive	`hdfs:///user/templeton/` `hcatalog-0.3.0.tar.gz`	The path to the Hive archive.
templeton.hive.path	`hcatalog-0.3.0.tar.gz/` `hcatalog-0.3.0/bin/hive`	The path to the Hive executable.

Name	Default	Description
templeton.streaming.jar	`hdfs:///user/templeton/` `hadoop-streaming.jar`	The path to the Hadoop streaming jar file.

1.3 Configuration

The configuration for Templeton merges the normal Hadoop configuration with the Templeton specific variables. Because Templeton is designed to connect services that are not normally connected, the configuration is more complex than might be desirable.

The Templeton specific configuration is split into three layers:

1. **templeton-default.xml** - All the configuration variables that Templeton needs. This file sets the defaults that ship with Templeton and should only be changed by Templeton developers. Do not copy this file and/or change it to maintain local installation settings. Because templeton-default.xml is present in the Templeton war file, editing a local copy of it will not change the configuration.
2. **templeton-dist.xml** - The (possibly empty) configuration file that can set variables for a particular distribution, such as an RPM file.
3. **templeton-site.xml** - The (possibly empty) configuration file in which the system administrator can set variables for their Hadoop cluster. Create this file and maintain entries in it for configuration variables that require you to override default values based on your local installation.

The configuration files are loaded in this order with later files overriding earlier ones.

Note: the Tomcat server will require restart after any change to the Templeton configuration.

To find the configuration files, Templeton first attempts to load a file from the CLASSPATH and then looks in the directory specified in the TEMPLETON_HOME environment variable.

Configuration files may access the special environment variable env for all environment variables. For example, the pig executable could be specified using:

```
${env.PIG_HOME}/bin/pig
```

Configuration variables that use a filesystem path try to have reasonable defaults. However, it's always safe to specify the full and complete path if there is any uncertainty.

1.3.1 Variables

Name	Default	Description
templeton.hadoop.config.dir	`$(env.HADOOP_CONFIG_DIR)`	The path to the Hadoop configuration.

Name	Default	Description
templeton.jar	`${env.TEMPLETON_HOME}/` `templeton/` `templeton-0.1.0-dev.jar`	The path to the Templeton jar file.
templeton.libjars	`${env.TEMPLETON_HOME}/` `lib/zookeeper-3.3.4.jar`	Jars to add to the classpath.
templeton.override.jars		Jars to add to the HADOOP_CLASSPATH for all Map Reduce jobs. These jars must exist on HDFS.
templeton.streaming.jar	`hdfs:///user/templeton/` `hadoop-streaming.jar`	The hdfs path to the Hadoop streaming jar file.
templeton.hadoop	`${env.HADOOP_PREFIX}/` `bin/hadoop`	The path to the Hadoop executable.
templeton.pig.archive	`hdfs:///user/templeton/` `pig-0.9.2.tar.gz`	The path to the Pig archive.
templeton.pig.path	`pig-0.9.2.tar.gz/` `pig-0.9.2/bin/pig`	The path to the Pig executable.
templeton.hcat	`${env.HCAT_PREFIX}/bin/` `hcat`	The path to the Hcatalog executable.
templeton.hive.archive	`hdfs:///user/templeton/` `hcatalog-0.3.0.tar.gz`	The path to the Hive archive.
templeton.hive.path	`hcatalog-0.3.0.tar.gz/` `hcatalog-0.3.0/bin/hive`	The path to the Hive executable.

templeton.hive.properties	`hive.metastore.local=fal`
	`hive.metastore.uris=thri`
	`localhost:9933,`
	`hive.metastore.sasl.enab`
		Properties to set when running hive.

templeton.exec.encoding	`UTF-8`	The encoding of the stdout and stderr data.
templeton.exec.timeout	`10000`	How long in milliseconds a program is allowed to run on the Templeton box.
templeton.exec.max-procs	`16`	The maximum number of processes allowed to run at once.

Name	Default	Description
templeton.exec.max-output-bytes	`1048576`	The maximum number of bytes from stdout or stderr stored in ram.
templeton.exec.envs	`HADOOP_PREFIX,HADOOP_HOM`	The environment variables passed through to exec.
templeton.zookeeper.hosts	`127.0.0.1:2181`	ZooKeeper servers, as comma separated host:port pairs
templeton.zookeeper.session-timeout	`30000`	ZooKeeper session timeout in milliseconds
templeton.callback.retry.interval	`10000`	How long to wait between callback retry attempts in milliseconds
templeton.callback.retry.attempts	`5`	How many times to retry the callback
templeton.zookeeper.cleanup.inter	`43200000`	The maximum delay between a thread's cleanup checks
templeton.zookeeper.cleanup.max	`604800000`	The maximum age of a templeton job

1.4 Reference

1.4.1 Templeton Resources

Resource	Description
status	Returns the Templeton server status.
ddl	Performs an HCatalog DDL command.
mapreduce/streaming	Creates and queues Hadoop streaming MapReduce jobs.
mapreduce/jar	Creates and queues standard Hadoop MapReduce jobs.
pig	Creates and queues Pig jobs.
hive	Runs Hive queries and commands.
queue	Returns a list of all jobids registered for the user.
GET queue/:jobid	Returns the status of a job given its ID.

Resource	Description
DELETE queue/:jobid	Kill a job given its ID.

1.4.2 GET status

1.4.2.1 Description

Returns the current status of the Templeton server. Useful for heartbeat monitoring.

1.4.2.2 URL

```
http://www.myserver.com/templeton/v1/status.json
```

1.4.2.3 Parameters

Only the standard parameters are accepted.

1.4.2.4 Results

Name	Description
status	"ok" if the Templeton server was contacted.
version	String containing the version number similar to "v1".

1.4.2.5 Example

Curl Command

```
% curl 'http://www.myserver.com/templeton/v1/status.json?user.name=charlotte'
```

JSON Output

```
{
 "status": "ok",
 "version": "v1"
}
```

1.4.3 POST ddl

1.4.3.1 Description

Performs an HCatalog DDL command. The command is executed immediately upon request. Responses are limited to 1MB. For requests which may return longer results consider using the Hive resource as an alternative.

1.4.3.2 URL

```
http://www.myserver.com/templeton/v1/ddl.json
```

1.4.3.3 Parameters

Name	Description	Required?	Default
exec	The HCatalog ddl string to execute	Required	None
group	The user group to use when creating a table	Optional	None
permissions	The permissions string to use when creating a table. The format is "`rwxrw-r-x`".	Optional	None

1.4.3.4 Results

Name	Description
stdout	A string containing the result HCatalog sent to standard out (possibly empty).
stderr	A string containing the result HCatalog sent to standard error (possibly empty).
exitcode	The exitcode HCatalog returned.

1.4.3.5 Example

Curl Command

```
% curl -d user.name=ctdean \
       -d 'exec=show tables;' \
       'http://www.myserver.com/templeton/v1/ddl.json'
```

JSON Output

```
{
 "stdout": "important_table
            my_other_table
            my_table
            my_table_2
            pokes
            ",
 "stderr": "WARNING: org.apache.hadoop.metrics.jvm.EventCounter is deprecated...
            Hive history file=/tmp/ctdean/hive_job_log_ctdean_201111111258_2117356679.txt
```

```
           OK
           Time taken: 1.202 seconds
           ",
 "exitcode": 0
}
```

1.4.4 POST mapreduce/streaming

1.4.4.1 Description

Create and queue an [Hadoop streaming MapReduce](#) job.

1.4.4.2 URL

`http://www.myserver.com/templeton/v1/mapreduce/streaming.json`

1.4.4.3 Parameters

Name	Description	Required?	Default
input	Location of the input data in Hadoop.	Required	None
output	Location in which to store the output data. If not specified, Templeton will store the output in a location that can be discovered using the [queue](#) resource.	Optional	See description
mapper	Location of the mapper program in Hadoop.	Required	None
reducer	Location of the reducer program in Hadoop.	Required	None
file	Add an HDFS file to the distributed cache.	Optional	None
define	Set an Hadoop configuration variable using the syntax `define=NAME=VALUE`	Optional	None
cmdenv	Set an environment variable using the syntax `cmdenv=NAME=VALUE`	Optional	None
arg	Set a program argument.	Optional	None

Name	Description	Required?	Default
statusdir	A directory where Templeton will write the status of the Map Reduce job. If provided, it is the caller's responsibility to remove this directory when done.	Optional	None
callback	Define a URL to be called upon job completion. You may embed a specific job ID into this URL using `$jobId`. This tag will be replaced in the callback URL with this job's job ID.	Optional	None

1.4.4.4 Results

Name	Description
id	A string containing the job ID similar to "job_201110132141_0001".
info	A JSON object containing the information returned when the job was queued. See the Hadoop documentation ([Class TaskController](#)) for more information.

1.4.4.5 Example

Code and Data Setup

```
% cat mydata/file01 mydata/file02
Hello World Bye World
Hello Hadoop Goodbye Hadoop

% hadoop fs -put mydata/ .

% hadoop fs -ls mydata
Found 2 items
-rw-r--r--   1 ctdean supergroup         23 2011-11-11 13:29 /user/ctdean/mydata/file01
-rw-r--r--   1 ctdean supergroup         28 2011-11-11 13:29 /user/ctdean/mydata/file02
```

Curl Command

```
% curl -d user.name=ctdean \
       -d input=mydata \
       -d output=mycounts \
       -d mapper=/bin/cat \
       -d reducer="/usr/bin/wc -w" \
       'http://www.myserver.com/templeton/v1/mapreduce/streaming.json'
```

JSON Output

```
{
 "id": "job_201111111311_0008",
 "info": {
          "stdout": "packageJobJar: [] [/Users/ctdean/var/hadoop/hadoop-0.20.205.0/share/
hadoop/contrib/streaming/hadoop-streaming-0.20.205.0.jar...
                    templeton-job-id:job_201111111311_0008
                    ",
          "stderr": "11/11/11 13:26:43 WARN mapred.JobClient: Use GenericOptionsParser for
 parsing the arguments
                    11/11/11 13:26:43 INFO mapred.FileInputFormat: Total input paths to
 process : 2
                    ",
          "exitcode": 0
         }
}
```

Results

```
% hadoop fs -ls mycounts
Found 3 items
-rw-r--r--   1 ctdean supergroup          0 2011-11-11 13:27 /user/ctdean/mycounts/_SUCCESS
drwxr-xr-x   - ctdean supergroup          0 2011-11-11 13:26 /user/ctdean/mycounts/_logs
-rw-r--r--   1 ctdean supergroup         10 2011-11-11 13:27 /user/ctdean/mycounts/
part-00000

% hadoop fs -cat mycounts/part-00000
      8
```

1.4.5 POST mapreduce/jar

1.4.5.1 Description

Creates and queues a standard Hadoop MapReduce job.

1.4.5.2 URL

```
http://www.myserver.com/templeton/v1/mapreduce/jar.json
```

1.4.5.3 Parameters

Name	Description	Required?	Default
jar	Name of the jar file for Map Reduce to use.	Required	None
class	Name of the class for Map Reduce to use.	Required	None
libjars	Comma separated jar files to include in the classpath.	Optional	None
files	Comma separated files to be copied to the map reduce cluster	Optional	None
arg	Set a program argument.	Optional	None
define	Set an Hadoop configuration variable using the syntax `define=NAME=VALUE`	Optional	None
statusdir	A directory where Templeton will write the status of the Map Reduce job. If provided, it is the caller's responsibility to remove this directory when done.	Optional	None
callback	Define a URL to be called upon job completion. You may embed a specific job ID into this URL using `$jobId`. This tag will be replaced in the callback URL with this job's job ID.	Optional	None

1.4.5.4 Results

Name	Description
id	A string containing the job ID similar to "job_201110132141_0001".

Name	Description
info	A JSON object containing the information returned when the job was queued. See the Hadoop documentation (Class TaskController) for more information.

1.4.5.5 Example

Code and Data Setup

```
% hadoop fs -put wordcount.jar .
% hadoop fs -put transform.jar .

% hadoop fs -ls .
Found 2 items
-rw-r--r--   1 ctdean supergroup        23 2011-11-11 13:29 /user/ctdean/wordcount.jar
-rw-r--r--   1 ctdean supergroup        28 2011-11-11 13:29 /user/ctdean/transform.jar
```

Curl Command

```
% curl -d user.name=ctdean \
       -d jar=wordcount.jar \
       -d class=org.myorg.WordCount \
       -d libjars=transform.jar \
       -d arg=wordcount/input \
       -d arg=wordcount/output \
       'http://www.myserver.com/templeton/v1/mapreduce/jar.json'
```

JSON Output

```
{
 "id": "job_201111121211_0001",
 "info": {
         "stdout": "templeton-job-id:job_201111121211_0001
                   ",
         "stderr": "",
         "exitcode": 0
         }
}
```

1.4.6 POST pig

1.4.6.1 Description

Create and queue a Pig job.

1.4.6.2 URL

```
http://www.myserver.com/templeton/v1/pig.json
```

1.4.6.3 Parameters

Name	Description	Required?	Default
execute	String containing an entire, short pig program to run.	One of either "execcute" or "file" is required	None
file	HDFS file name of a pig program to run.	One of either "exec" or "file" is required	None
arg	Set a program argument.	Optional	None
files	Comma separated files to be copied to the map reduce cluster	Optional	None
statusdir	A directory where Templeton will write the status of the Pig job. If provided, it is the caller's responsibility to remove this directory when done.	Optional	None
callback	Define a URL to be called upon job completion. You may embed a specific job ID into this URL using $jobId. This tag will be replaced in the callback URL with this job's job ID.	Optional	None

1.4.6.4 Results

Name	Description
id	A string containing the job ID similar to "job_201110132141_0001".
info	A JSON object containing the information returned when the job was queued. See the Hadoop documentation (Class TaskController) for more information.

1.4.6.5 Example

Code and Data Setup

```
% cat id.pig
A = load 'passwd' using PigStorage(':');
B = foreach A generate $0 as id;
dump B;

% cat fake-passwd
ctdean:Chris Dean:secret
pauls:Paul Stolorz:good
carmas:Carlos Armas:evil
dra:Deirdre McClure:marvelous

% hadoop fs -put id.pig .
% hadoop fs -put fake-passwd passwd
```

Curl Command

```
% curl -d user.name=ctdean \
       -d file=id.pig \
       -d arg=-v \
       'http://www.myserver.com/templeton/v1/pig.json'
```

JSON Output

```
{
 "id": "job_201111101627_0018",
 "info": {
         "stdout": "templeton-job-id:job_201111101627_0018
                   ",
         "stderr": "",
         "exitcode": 0
       }
}
```

1.4.7 POST hive

1.4.7.1 Description

Runs a [Hive](#) query or set of commands.

1.4.7.2 URL

```
http://www.myserver.com/templeton/v1/hive.json
```

1.4.7.3 Parameters

Name	Description	Required?	Default
execute	String containing an entire, short hive program to run.	One of either "execute" or "file" is required	None

Name	Description	Required?	Default
file	HDFS file name of a hive program to run.	One of either "exec" or "file" is required	None
define	Set a Hive configuration variable using the syntax define=NAME=VALUE.	Optional	None
statusdir	A directory where Templeton will write the status of the Hive job. If provided, it is the caller's responsibility to remove this directory when done.	Optional	None
callback	Define a URL to be called upon job completion. You may embed a specific job ID into this URL using $jobId. This tag will be replaced in the callback URL with this job's job ID.	Optional	None

1.4.7.4 Results

Name	Description
id	A string containing the job ID similar to "job_201110132141_0001".
info	A JSON object containing the information returned when the job was queued. See the Hadoop documentation ([Class TaskController](#)) for more information.

1.4.7.5 Example

Curl Command

```
% curl -d user.name=ctdean \
       -d execute="select+*+from+pokes;" \
       -d statusdir="pokes.output" \
       'http://www.myserver.com/templeton/v1/hive.json'
```

JSON Output

```
{
 "id": "job_201111111311_0005",
 "info": {
          "stdout": "templeton-job-id:job_201111111311_0005
                    ",
          "stderr": "",
          "exitcode": 0
         }
}
```

Results

```
% hadoop fs -ls pokes.output
Found 2 items
-rw-r--r--   1 ctdean supergroup        610 2011-11-11 13:22 /user/ctdean/pokes.output/
stderr
-rw-r--r--   1 ctdean supergroup         15 2011-11-11 13:22 /user/ctdean/pokes.output/
stdout

% hadoop fs -cat pokes.output/stdout
1       a
2       bb
3       ccc
```

1.4.8 GET queue

1.4.8.1 Description

Return a list of all job IDs registered to the user.

1.4.8.2 URL

`http://www.myserver.com/templeton/v1/queue.json`

1.4.8.3 Parameters

Only the standard parameters are accepted.

1.4.8.4 Results

Name	Description
ids	A list of all job IDs registered to the user.

1.4.8.5 Example

Curl Command

```
% curl 'http://www.myserver.com/templeton/v1/queue.json?user.name=ctdean'
```

JSON Output

```
{
 "job_201111111311_0008",
 "job_201111111311_0012"
}
```

1.4.9 GET queue/:jobid

1.4.9.1 Description

Check the status of a job and get related job information given its job ID. Substitute ":jobid" with the job ID received when the job was created.

1.4.9.2 URL

`http://www.myserver.com/templeton/v1/queue/:jobid.json`

1.4.9.3 Parameters

Name	Description	Required?	Default
:jobid	The job ID to check. This is the ID received when the job was created.	Required	None

1.4.9.4 Results

Name	Description
status	A JSON object containing the job status information. See the Hadoop documentation ([Class JobStatus](#)) for more information.
profile	A JSON object containing the job profile information. See the Hadoop documentation ([Class JobProfile](#)) for more information.
id	The job ID.
parentId	The parent job ID.
percentComplete	The job completion percentage, for example "75% complete".
exitValue	The job's exit value.
user	User name of the job creator.

Name	Description
callback	The callback URL, if any.
completed	A string representing completed status, for example "done".

1.4.9.5 Example

Curl Command

```
% curl 'http://www.myserver.com/templeton/v1/queue/job_201112212038_0003.json?
user.name=ctdean'
```

JSON Output

```
{
 "status": {
            "startTime": 1324529476131,
            "username": "ctdean",
            "jobID": {
                       "jtIdentifier": "201112212038",
                       "id": 4
                     },
            "jobACLs": {
                       },
            "schedulingInfo": "NA",
            "failureInfo": "NA",
            "jobId": "job_201112212038_0004",
            "jobPriority": "NORMAL",
            "runState": 2,
            "jobComplete": true
          },
 "profile": {
             "url": "http://localhost:50030/jobdetails.jsp?jobid=job_201112212038_0004",
             "jobID": {
                        "jtIdentifier": "201112212038",
                        "id": 4
                      },
             "user": "ctdean",
             "queueName": "default",
             "jobFile": "hdfs://localhost:9000/tmp/hadoop-ctdean/mapred/staging/
ctdean/.staging/job_201112212038_0004/job.xml",
             "jobName": "PigLatin:DefaultJobName",
             "jobId": "job_201112212038_0004"
           },
 "id": "job_201112212038_0004",
 "parentId": "job_201112212038_0003",
 "percentComplete": "100% complete",
 "exitValue": 0,
 "user": "ctdean",
 "callback": null,
 "completed": "done"
}
```

1.4.10 DELETE queue/:jobid

1.4.10.1 Description

Kill a job given its job ID. Substitute ":jobid" with the job ID received when the job was created.

1.4.10.2 URL

```
http://www.myserver.com/templeton/v1/queue/:jobid.json
```

1.4.10.3 Parameters

Name	Description	Required?	Default
:jobid	The job ID to delete. This is the ID received when the job job was created.	Required	None

1.4.10.4 Results

Name	Description
status	A JSON object containing the job status information. See the Hadoop documentation (Class JobStatus) for more information.
profile	A JSON object containing the job profile information. See the Hadoop documentation (Class JobProfile) for more information.
id	The job ID.
parentId	The parent job ID.
percentComplete	The job completion percentage, for example "75% complete".
exitValue	The job's exit value.
user	User name of the job creator.
callback	The callback URL, if any.
completed	A string representing completed status, for example "done".

1.4.10.5 Example

Curl Command

```
% curl -X DELETE 'http://www.myserver.com/templeton/v1/queue/job_201111111311_0009.json?
user.name=ctdean'
```

JSON Output

```
{
 "status": {
            "startTime": 1321047216471,
            "username": "ctdean",
            "jobID": {
                        "jtIdentifier": "201111111311",
                        "id": 9
                     },
            "jobACLs": {
                        },
            "schedulingInfo": "NA",
            "failureInfo": "NA",
            "jobId": "job_201111111311_0009",
            "jobPriority": "NORMAL",
            "runState": 1,
            "jobComplete": false
          },
 "profile": {
            "url": "http://localhost:50030/jobdetails.jsp?jobid=job_201111111311_0009",
            "user": "ctdean",
            "jobID": {
                        "jtIdentifier": "201111111311",
                        "id": 9
                     },
            "queueName": "default",
            "jobFile": "hdfs://localhost:9000/tmp/hadoop-ctdean/mapred/staging/
ctdean/.staging/job_201111111311_0009/job.xml",
            "jobName": "streamjob3322518350676530377.jar",
            "jobId": "job_201111111311_0009"
          }
 "id": "job_201111111311_0009",
 "parentId": "job_201111111311_0008",
 "percentComplete": "10% complete",
 "exitValue": 0,
 "user": "ctdean",
 "callback": null,
 "completed": "false"
}
```

Note: The job is not immediately deleted, therefore the information returned may not reflect deletion, as in our example. Use GET queue/:jobid to monitor the job and confirm that it is eventually deleted.

1.5 PDF