USER GUIDE FOR ABCGrid

Last update: Feb-07-2006, version: 0.1

Ying Sun
Center for Bioinformatics
College of Life Sciences
Peking University
Beijing 100871, P.R.China
suny@mail.cbi.pku.edu.cn

ABCGrid is a server/client application consists of three independent applications: ABCMaster (manage job I/O), ABCWorker(process job) and ABCUser(send commands). The machine runs ABCMaster is called "master node", the machine runs ABCWorker is called "worker node", the machine runs ABCUser is called "user node". In one typical environment, there are one master node and some worker nodes connected by local area network:

```
                                                  |---Worker_A
User_A----|                                       |---Worker_B
          |                                       |---Worker_C
User_B----|                           intranet    |---Worker_D
          |←----------→Master←------------→|---Worker_E
User_C----|                                       |---Worker_F
          |                                       |---Worker_G
User_…    |                                       |---Worker_H
                                                  |---Worker_I
                                                  |---Worker_…
```

These nodes are connected by intranet(Theoretically, ABCGrid supports internet, but we did not implement high security policies except for password and IP access authentication in this version of ABCGrid. ABCGrid is intended to be used in small labs with intranet and little security issues. However, existing username/password and IP access restrictions are enough to block most hackers). Master node and worker nodes are loosely coupled, machines can join the computing grid (by connecting to master node) at any time as worker nodes.

The quickest way to install and run

Before you install ABCGrid, please make sure you have JRE1.5 (http://java.sun.com/javase/) or above installed. Use command "/JAVA_HOME/bin/java -version" to see current version.

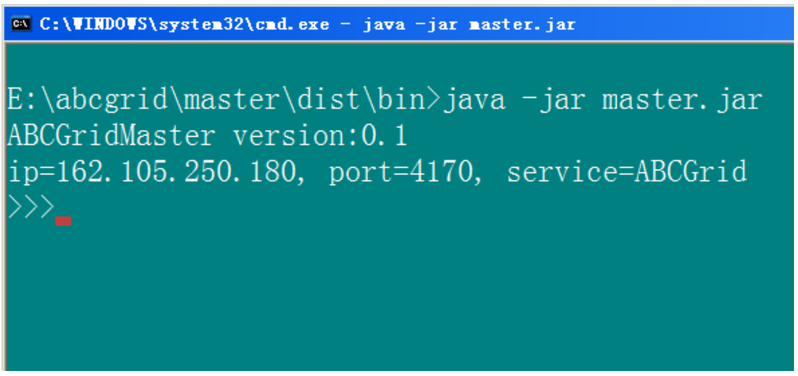
You can install all the three applications into one machine to try to run them (If your machine has firewall installed, please disable it temporarily or open port 4170 and 4171 for ABCGrid). For demonstration, ABCUser is not used.

1) **Downloaded master-bin.zip and worker-bin.zip**

2) **Unzip master-bin.zip and worker-bin.zip using WinZip, WinRAR (Windows) or unzip (UNIX/Linux)**
   Suppose you have unzipped them to c:\master and c:\worker respectively

3) **Open two command windows**
   In Windows, click "start"->"run", input "cmd" to start a command window. You should create two command windows for ABCMaster and ABCWorker respectively.

4) **Run ABCMaster**
   Change to the first command window.
   >cd c:\master\bin
   >java -jar master.jar
   If everything is OK, a console windows appears:



```
C:\WINDOWS\system32\cmd.exe - java -jar master.jar

E:\abcgrid\master\dist\bin>java -jar master.jar
ABCGridMaster version:0.1
ip=162.105.250.180, port=4170, service=ABCGrid
>>>_
```

   The line: "**ip=162.105.250.180, port=4170, service=ABCGrid**" specified the master node's address (in my machine). The IP should be different in your machine. The IP address is required when start ABCWorker since it must be told which master to connect with.

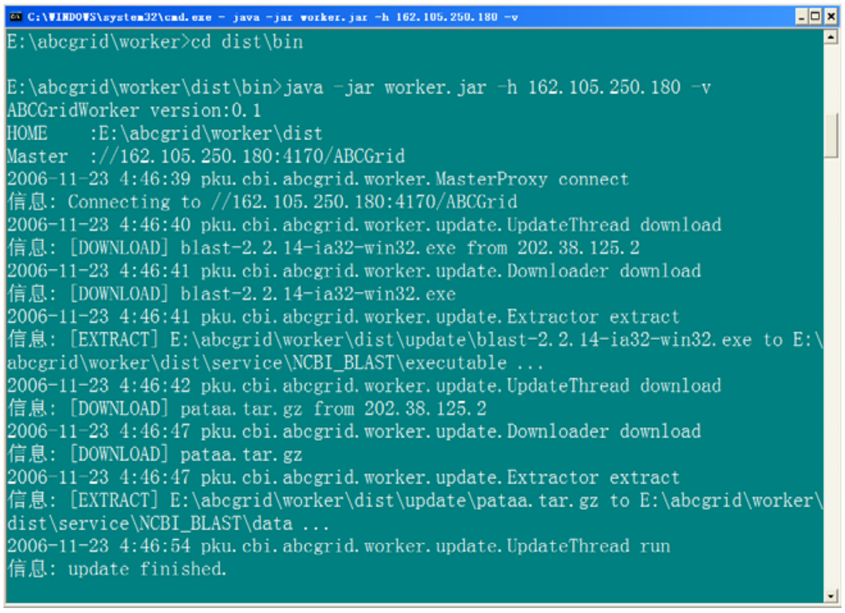
5) **Run ABCWorker**
   Switch to the second command window.
   >cd c:\worker\bin
   >java -jar worker.jar -h 162.105.250.180 -v
   The parameters "***-h 162.105.250.180***" is used to specify that the host(master)'s address is "162.105.250.180". Change this value to your machine's IP address. The "-h xxx.xxx.xxx.xxx" is only needed when you start the ABCWorker for the first time or you changed the

master's address since last launch of ABCWorker. If master's address is not changed, just type:

>>>java -jar worker.jar

is OK. The parameter "**-v**" is "verbose" mode.

If everything is OK, the Worker will download and update local applications and databases. You will see a console windows:



```
C:\WINDOWS\system32\cmd.exe - java -jar worker.jar -h 162.105.250.180 -v
E:\abcgrid\worker>cd dist\bin

E:\abcgrid\worker\dist\bin>java -jar worker.jar -h 162.105.250.180 -v
ABCGridWorker version:0.1
HOME    :E:\abcgrid\worker\dist
Master  ://162.105.250.180:4170/ABCGrid
2006-11-23 4:46:39 pku.cbi.abcgrid.worker.MasterProxy connect
信息: Connecting to //162.105.250.180:4170/ABCGrid
2006-11-23 4:46:40 pku.cbi.abcgrid.worker.update.UpdateThread download
信息: [DOWNLOAD] blast-2.2.14-ia32-win32.exe from 202.38.125.2
2006-11-23 4:46:41 pku.cbi.abcgrid.worker.update.Downloader download
信息: [DOWNLOAD] blast-2.2.14-ia32-win32.exe
2006-11-23 4:46:41 pku.cbi.abcgrid.worker.update.Extractor extract
信息: [EXTRACT] E:\abcgrid\worker\dist\update\blast-2.2.14-ia32-win32.exe to E:\
abcgrid\worker\dist\service\NCBI_BLAST\executable ...
2006-11-23 4:46:42 pku.cbi.abcgrid.worker.update.UpdateThread download
信息: [DOWNLOAD] pataa.tar.gz from 202.38.125.2
2006-11-23 4:46:47 pku.cbi.abcgrid.worker.update.Downloader download
信息: [DOWNLOAD] pataa.tar.gz
2006-11-23 4:46:47 pku.cbi.abcgrid.worker.update.Extractor extract
信息: [EXTRACT] E:\abcgrid\worker\dist\update\pataa.tar.gz to E:\abcgrid\worker\
dist\service\NCBI_BLAST\data ...
2006-11-23 4:46:54 pku.cbi.abcgrid.worker.update.UpdateThread run
信息: update finished.
```

Wait until the updating process is over. The last line with "update finished" denotes that the updating process is over and the ABCWorker is ready to accept jobs. If it failed and print a message like "error to connect ftp:xxxxx", that is because the connection is not established between ABCWorker and FTP server. In a default configuation, ABCGrid use ftp://202.38.125.2 to download all necessary files. For your convinience, you should setup a local FTP server to store all databases and executables.

6) **OK, run a job now.**

Switch to the ABCMaster's console window. Input commands after the ">>>" prompter.

>>>r blastall -p blastp -i pdb_10.fasta -d pataa -o pdb_10.blastp -m 9

If everything is OK, you will see a messages says your job is accepted with an automated assigned integer value begins from 1.

This command will do a **blastp** search using **pdb_10.fasta** (under c:\master\user\admin\) against database **pataa** (under c:\worker\service\NCBI_BLAST\data\). Once finished, the output file **pdb_10.blastp** can be found under c:\master\user\admin\. It is the working directory of user "admin".

During the computation, you can query the status of submitted job by using command "**j**"

```
>>>j
```

The fourth column with name “finished” shows the progress. If it was “100%”, the computation is over. Check the output file.

By default, ABCGrid defined two users: “demo” and “admin”. User who operates in the console window of ABCMaster is considered as “admin” automatically.

The normal way to install, configure and run

Binary installation for ABCMaster

1. Downloaded and unzip master-bin.zip

   ```
   >cd C:\master\
   >unzip master-bin.zip
   ```

2. Create a working directory for each user

   For example, create a directory for user “john” and “kate”:

   ```
   >cd C:\master\user
   >mkdir john
   >mkdir kate
   ```

   Directory C:\master\user\john will be input/output directory of user john. Input files must be placed under home directory before submitting jobs. Output files will be placed under this directory by ABCMaster, too. You can specify other directory other than under C:\master\user as one user’s working directory. See C:\master\conf\master.conf for details.

3. Run ABCMaster

   ```
   >cd C:\master\bin
   >java –jar master.jar
   ```

Source installation for ABCMaster

1. Download and unzip master-src.zip

   ```
   >cd C:\master
   >unzip master-src.zip
   ```

2. Build binary package

   ```
   >cd C:\master\
   >\home\to\apache-ant\bin\ant
   ```

   The build process should begin. If successful, a sub directory of "dist" will be created.

3. Run ABCMaster

   ```
   >cd c:\master\dist\bin
   >java –jar master.jar
   ```

Binary installation for ABCWorker

1. Download and unzip worker-bin.zip
```
>cd C:\worker
>unzip worker-bin.zip
```
2. Run ABCWorker
```
>cd C:\worker\bin
>java -jar worker.jar [-h IP_OF_MASTER] [-p PORT_OF_MASTER] [-v]
```

Source installation for ABCWorker

1. Download and unzip worker-src.zip
```
>cd C:\worker
>unzip worker-src.zip
```
2. Build binary package
```
>cd C:\worker\
>\home\to\apache-ant\bin\ant
```
The build process should begin. If successful, a sub directory of "dist" will be created.
3. Run ABCWorker
```
>cd c:\worker\dist\bin
> java -jar worker.jar [-h IP_OF_MASTER] [-p PORT_OF_MASTER] [-v]
```

Binary installation for ABCUser

1. Download and unzip user-bin.zip
```
>cd C:\user
>unzip user-bin.zip
```
2. Run ABCUser
```
>cd C:\user
>java -jar user.jar [command]
```

Source installation for ABCUser

1. Downloaded and unzip user-src.zip
```
>cd C:\user
>unzip user-src.zip
```
2. Build binary package
```
>cd C:\user\
>\home\to\apache-ant\bin\ant
```
3. Run ABCUser
```
>cd C:\user\dist\bin
>java -jar user.jar [command]
```

CONFIGURE

The configuration can be divided into three parts:

1. ***master.conf*** for configuration of Master's address and User control. -> C:\master\conf\master.conf
2. ***update.conf*** for online updating -> C:\master\conf\update.conf
3. ***user.conf*** for User's identity -> C:\user\conf\user.conf

Master.conf defined Master's address and user list. Address is used to tell Worker and User the connecting ports. User is used to control all grid users. Each grid user has his six attributes: name, password, address, priority, group and home. The first three were used to authenticate user. Priority defined the job's priority. Job with higher priority is processed early(Internally we use a priority queue). Group defines what kind of operation of one User could do. Some operations such as updating could only be executed by administrator. The last attribute "home" is the user's working directory. When a user submit a job, Master depends on this value to find input files and put output files.

ABCGrid can work without ABCUser. In this situation, you can submit jobs by using the ABCMaster's console window alone. Note the user who operate on the Console windows is considered as "admin" which
has the highest privilege: updating, kill anyone's job, etc.

ABCGrid use JavaRMI as the communication protocol between Master and Worker, Socket as the communication protocol between Master and User.

```
          Socket                              JavaRMI
User←--------------------→Master←---------------------→Worker
```

User and Worker must know the Master's IP address and port number to connect with. By default, Master will use port 4170 for Master-Worker and port 4171 for Master-User.

Suppose the Master machine's IP address is "12.34.56.78", and we want to change the Master-Worker port from 4170 to 8192, Master-User port from 4170 to 9999.

In Master machine:

1. Open file "master.conf" in c:\master\conf

2. change line 5 from “**<port>4170</port>**” to “**<port>8192</port>**”

In Worker machine:

After you changed port number in Master and started Master, run Worker:

```
>java –jar worker.jar –h 12.34.56.78 –p 8192
```

This command will write Master’s address information to local configuration file “worker.conf”. If the Master’s address does not change, next time we just run

```
>java –jar worker.jar
```

will connect to 12.34.56.78:8192 by default, no need to specify Master’s address again as long as Master’s address does not changed.

In User machine:

After you changed port number in Master and started Master, open the file “user.conf” under c:\user\conf and change the following two lines with:

```
master=12.34.56.78
port=8192
```

Online Update configuarion: update.conf

Attention: This is the most complicated part of ABCGrid, read it carefully and make sure you understand before you start configure the online update.

The central idea behind the online update is: Master keeps a file(c:\master\conf\update.conf) contains the information on all data source, computing platforms, necessary applications and databases. When Worker connected to Master, the first think to do is get a copy of this information from Master, compare it with local configuration file, find differences, download necessary files from specified FTP server, extract downloaded files and last, update local configuration file(c:\worker\conf\worker.conf).

First, you should know how “update.conf” is organized. “update.conf” is a XML-formatted file which is defined by a XSD file(update.xsd). You can open “update.conf” with any text editor. For convenience, we put a demo of this file here:

```
<xml version=”1.0”>
<grid>
    <!–list of ftp servers which contain files for updating-->
```

```
<ftplist>
    <ftp name="ncbi_blast_exec">
        <ip>ftp.ncbi.nlm.nih.gov</ip>
        <port>21</port>
        <user>anonymous</user>
        <password>anonymous</password>
        <!—subdirectory which contains the target file-->
        <basepath>blast/executables/LATEST/</basepath>
        <pasv>1</pasv>
    </ftp>
    <ftp name="ncbi_blast_data">
        <ip>ftp.ncbi.nlm.nih.gov</ip>
        <port>21</port>
        <user>anonymous</user>
        <password>anonymous</password>
        <basepath>blast/db/</basepath>
        <pasv>1</pasv>
    </ftp>
<ftplist>

<!--for i386 machines running linux-->
<services os="Linux" arch="i386" >
        <service name="NCBI_BLAST">
            <!—-executable component-->
            <component type="executable" ftp="ncbi_blast_exec">
                <name>blast-2.2.14-ia32-linux.tar.gz</name>
                <path>blast-2.2.14/bin</path>
            </component>
            <!—-data component-->
            <component type="data" ftp="ncbi_blast_data">
                <name>nr.tar.gz</name>
                <path></path>
            </component>
            <component type="data" ftp="ncbi_blast_data">
                <name>pataa.tar.gz</name>
                <path></path>
            </component>
        </service>

        <service name="HMMER">
            <component type="executable" ftp="suny_hmmer_exec">
                <name>hmmer-2.3.2-x86-linux.tar.gz</name>
                <path>bin</path>
            </component>
```

```
                <component type="data" ftp="suny_hmmer_db">
                    <name>Pfam_ls.gz</name>
                    <path></path>
                </component>
            </service>
    </services>
    <!--for x86 machines running windows-->
    <services os="Windows" arch="x86">
        <service name="NCBI_BLAST">
            <component type="executable" ftp=" ncbi_blast_exec">
                <name>blast-2.2.14-ia32-win32.exe</name>
                <path>bin/</path>
            </component>
            <component type="data" ftp="ncbi_blast_data">
                <name>nr.tar.gz</name>
                <path></path>
            </component>
            <component type="data" ftp=" ncbi_blast_data">
                <name>pataa.tar.gz</name>
                <path></path>
            </component>
       </service>
    </services>
</grid>
```

What did this file tell us?

1. Two ftp server, one is **ncbi_blast_exec**, another is **ncbi_blast_data**.
2. Support two computing platform, Linux/i386 and Windows/x86.
3. For Linux/i386 platform, two services should be supported. NCBI_BLAST and HMMER. Service NCBI_BLAST has three components: one executable component including all executable files such as blastall, megablast and two data components including nr database and pataa database. Service HMMER has two components: one executable component including executable files such as hmmpfam and one data component Pfam_ls database. Note all databases must be pre-formatted.
4. For Windows/x86 platform, only one services NCBI_BLAST should be supported. Service NCBI_BLAST has three components: one executable component and two data components same as the Linux/i386 platform. **ATTENTION: One service MUST has same components in all platforms. ABCWorker get tasks based on its supported services not on detailed service components. If you ask a blast-against-nr operation, Worker will take this job even it does not install nr database. Consequently this job is discarded at worker node, no results could be returned.**

Now let's go into more details. **<service>** is the basic element which defined an application package, such as NCBI_BLAST and HMMER. Service is consists of **<components>** which could be characterized by two categories: ***executable component*** and ***data component***. Take NCBI_BLAST as example, blastall, rpsblast, bl2seq both belong to executable component. Pre-formatted nr, nt and swissprot both belong to data component. One **service** MUST has at least one executable component. We use an example to illustrate the details line by line.

```
1.  <services os="Linux" arch="i386" >
2.      <service name="NCBI_BLAST">
3.          <component type="executable" ftp="ncbi_blast_exec">
4.              <name>blast-2.2.14-ia32-linux.tar.gz</name>
5.              <path>blast-2.2.14/bin</path>
6.          </component>
7.          <component type="data" ftp="ncbi_blast_data">
8.              <name>nr.tar.gz</name>
9.              <path></path>
10.         </component>
11.         <component type="data" ftp="ncbi_blast_data">
12.             <name>pataa.tar.gz</name>
13.             <path></path>
14.         </component>
15.     </service>
16. </services>
```

Line 1. <services os="Linux" arch="i386" >
These services are intended to use in i386 machine running Linux.

Line 2. <service name="NCBI_BLAST">
The service is NCBI_BLAST.

Line 3,4,5,6

```
<component type="executable" ftp="ncbi_blast_exec">
    <name>blast-2.2.14-ia32-linux.tar.gz</name>
    <path>blast-2.2.14/bin</path>
</component>
```

***This paragraph is very important.**
This section defined the *executable* component of NCBI_BLAST. This component should be downloaded from "**ftp:ncbi_blast_exec**" which is defined by above **<ftp>** entry. The component's file name is blast-2.2.14-ia32-linux.tar.gz. After being downloaded and extracted, this component can be found under
C:\worker\service\NCBI_BLAST\executable\blast-2.2.14\bin

***Suppose you have installed Worker application under C:\worker\**

Where is the path coming from? After download and extract one component, the Worker must know where to find the extracted component which could be either executable applications or database. Worker will download blast-2.2.14-ia32-linux.tar.gz and extract it to

C:\worker\service\$SERVICE_NAME\$COMPONENT_TYPE\$COMPONENT_PATH.

In this example, $SERVICE_NAME is "NCBI_BLAST", $COMPONENT_TYPE is "executable" and $COMPONENT_PATH is "blast-2.2.14/bin". So the full path to extracted blast-2.2.14-ia32-linux.tar.gz is

C:\worker\service\NCBI_BLAST\executable\blast-2.2.14\bin

List the files in that directory, you will find all NCBI_BLAST executable files such as "blastall" and "rpsblast" etc.

Why the path must be defined as <path>blast-2.2.14/bin</path>? Because extracted blast-2.2.14-ia32-linux.tar.gz will make some sub-directories.

```
>cd demo
>tar -zxvf blast-2.2.14-ia32-linux.tar.gz
```

The directory tree looks like:

```
  demo/
    +---blast-2.2.14/
              +----bin/
                     +---blastall*
                     +---bl2seq*
                     +--- ...
              +----doc/
              +----data/
```

The rule is also applied to data component. Line 7,8,9,10 defined a Data component of NCBI_BLAST:

```
    <component type="data" ftp="ncbi_blast_data">
        <name>nr.tar.gz</name>
        <path></path>
    </component>
```

Notice the component_path is blank, because extracted nr.tar.gz does not make any sub-directories. Now the nr database could be found under **C:\worker\service\NCBI_BLAST\data\**

To emphasis again, a component will be extracted to

C:\worker\service\$SERVICE_NAME\$COMPONENT_TYPE\$COMPONENT_PATH.

The windows version of NCBI_BLAST is .exe file. Execution of blast-2.2.14.exe

```
>cd c:\demo
>blast-2.2.14.exe
```
will make a directoy tree like:
```
  c:\demo\
    |
    +----bin\
           |
           +---blastall*
           +---bl2seq*
           +--- ...
    +----doc\
    +----data\
```

So the NCBI_BLAST executable component for Windows is:
```
    <component type="executable" ftp=" ncbi_blast_exec">
           <name>blast-2.2.14-ia32-win32.exe</name>
           <path>bin/</path>
    </component>
```
Make sure you set the right value of <path>. It is different from the Linux/i386 version.

The update process is launched from Master by input comamnd "**u**" in the console window. Once launched, the update process will try to find a configuration file $ABCGRID_MASTER_HOME/conf/update.conf and read its content, transfer the content data to all connected Worker nodes. Worker nodes will compare local copy with the update data from Master. If differences were found, e.g. add/delete and update.
The worker node will try to download data from ftp server and extracted to local disk, update the local configuration file.
So the very first thing to update all worker nodes is to configure the "update.conf" in Master node.

Note:

It is strongly recommended that all required data and applications from the original ftp site should be downloaded to a local ftp site. Local ftp site is generally much faster than original ftp site and can save lots of time.

COMMANDS

Commands that are supported by ABCUser and ABCMaster's console.
If you use ABCUser, the command should be:

/JAVA_HOME/bin/java -jar user.jar [command]

If you use ABCMaster's console window:

>>>[command]

We give examples by using ABCMaster's console window.

Run

Run a job. Before run a job, a User must put all input files in his home (working directory)

SYNTAX

r command [params]

EXAMPLE

>>>r blastall -p blastp -i pdb_100.fasta -d nr -o pdb_100.fasta.blastp -m 9

will run a blastp search again nr database using input sequence file pdb_100.fasta.

>>>r CE - pdbdemo1.ent A pdbdemo2.ent A scratch

will run a CE structure alignment between chain A of pdbdemo1.ent and chain B of pdbdemo2.ent.

Output and error redirection are also supported:

>>>r CE - pdbdemo1.ent A pdbdemo2.ent A scratch 1>abc

will redirect output to file "abc"

>>>r hmmpfam Pfam_ls input.seq >abc

will redirect output and error to file "abc"

>>>r hmmpfam Pfam_ls input.seq 1>abc 2>def

will redirect output to file "abc" and error to file "def". By default, there is a user "demo" with some files(You can find a "demo" directory under the "user" directory in Master) include some FASTA files and two PDB files. You can use these files to test the working environment. Besides FASTA, some other popular sequence file formats include **EMBL**, **Genbank**, **SwissProt** and **Pfam** are also supported as the input files.

Service

Show supported services.

SYNTAX

s

EXAMPLE

>>>s

show all supported services.

Job

Show the information of recent jobs.

SYNTAX

j   [jobid]

EXAMPLE

>>>j

print information of all jobs(which submitted by current user)

>>>j 1

print information of job with id=1.

>>>j 1 2 5 7

print information of jobs with id=1,2,5 and 7.

Worker

Show basic information of connected worker nodes including machine name, IP, operating system, architecture, current status(busy, idle, etc.)

SYNTAX

w

EXAMPLE

>>**>w**

print information of all connected worker nodes.

Kill

Kill a job. A User can only kill his jobs. If a User is administrator (which group is “admin”), he can kill any jobs.

SYNTAX

k jod_id

EXAMPLE

>>>k 1

kill the job which has id = 1

```
>>>k 0
```

kill all jobs submitted by current user. value 0 indicate all jobs.

Update

Update all connected worker nodes. This is a admin-only operation.

SYNTAX

u

EXAMPLE

>>>u

will initialize the update process.

Help

Show help message.

SYNTAX

h

EXAMPLE

>>>h

show help message.

Quit

Quit the Master application. This is a admin-only operation.

SYNTAX

q

EXAMPLE

>>>q

quit the Master application.

Note: This command can only terminate the master application. Worker application will NOT exist and continue try to connect to Master(run as a daemon app). If the Master was started again, connections between Master and Worker will be recovered. No need to start each Worker application running on Worker nodes one by one.