

# Distributed backup service

---

## How to run

From the Project's root folder:

1. To compile in LINUX/UNIX: using a terminal, navigate to the project's root folder and run the 'compile.sh' script:

```
sh compile.sh
```

2. To start the RMI registry run the 'rmi.sh' script:

```
sh rmi.sh
```

3. To start a peer, use the 'peer.sh' script:

```
sh peer.sh <protocol_version> <peer_id>
```

(e.g. `sh peer.sh 2.0 1`)

4. To run the TestApp:

- open a terminal, navigate to the project's root folder and type "sh <test\_action>.sh" - <test\_action> is one of **backup**, **restore**, **reclaim**, **delete** or **state**.
- To clear the peers' file system, run the 'clearFilesystem.sh' script:

```
sh clearFilesystem.sh
```

## How to compile

The project can be compiled as usual with the `javac` command, or by running `sh compile.sh` from the project's root folder.

## RMI registry

As suggested, the interface implementation uses RMI. In order to interact with the service, an *rmiregistry* instance must be running inside **bin** folder.

The following command launches an *rmiregistry* instance in the background: `rmiregistry &`

## Peer

If the user intends to interact with a peer, a name for the remote object must be specified, hence the alternative usages:

To run a peer:

```
Usage: java -classpath bin service.Peer <protocol_version> <server_id>  
<service_access_point> <mc:port> <mdb:port> <mdr:port>
```

**Argument description:**

protocol\_version – version of protocol used (1.0, 1.1, 1.2, 1.3 or 2.0)  
 server\_id – Integer representing the peer's unique identifier  
 service\_access\_point – host to access the rmi registry (format explained on RMI Registry section)  
 mc:port – IP address:Port of the CONTROL channel  
 mdb:port – IP address:Port of the BACKUP channel  
 mdr:port – IP address:Port of the RESTORE channel

**Protocol version:**

1.0 -> Default

//host(:port)?/

1.1 -> Backup Enhancement

//host(:port)?/peerID

1.2 -> Restore Enhancement

1.3 -> Delete Enhancement

2.0 -> All Enhancement

**service\_access\_point:**

– Formats on Peer:

– Formats on testApp:

Eg. java -classpath bin service.Peer "\$1" "\$2" //localhost/ 224.0.0.0:8000  
224.0.0.0:8001 224.0.0.0:8002

## Test App

To run test app

Usage: java TestApp <peer\_ap> <sub\_protocol> <opnd\_1> <opnd\_2>

**Argument description:**

peer\_app – local peer access point

sub-protocol – can be BACKUP, RESTORE, RECLAIM, DELETE or

STATE

opnd\_1 – path name of the file or amount of space to be

reclaim

opnd\_2 – specifies the desired replication degree. Applies

only to backup sub-protocol

Eg.

java -classpath bin service.TestApp //localhost/1 BACKUP

"files/image1.png" 1

java -classpath bin service.TestApp //localhost/1 DELETE

"files/image1.png"

java -classpath bin service.TestApp //localhost/2 RECLAIM 140000

java -classpath bin service.TestApp //localhost/1 RESTORE

"files/image1.png"

java -classpath bin service.TestApp //localhost/1 STATE

## Defaults

Usages where the user does not specify the multicast addresses and ports will have the following defaults (on script):

RMI	MC	MDB	MDR
//localhost:1099/	224.0.0.0:8000	224.0.0.0:8001	224.0.0.0:8002

The default size of a peer on system is 8MB.

## Local files and configuration

- The source files are under the project **src** folder.
- The class files are under the project **bin** folder.
- The files used to test are under the project **files** folder.
- The filesystem of each peer are under the project **fileSystem** folder, on each peer contains a **peerID** folder that contains **chunks** folder, **restore** folder and **db** file. Each **chunks** folder contains folders named by fileID that contains chunks backed up which belongs to that file.

You need not generate these files manually. Once the *Peer* is launched for the very first time, they will be automatically created.

You should not manually edit **db**. These files are managed by the service.

## Scripts' Specification

- compile.sh
  - Script to compile java classes into a bin folder.
- clearFileSystem.sh
  - Script to delete fileSystem of peers.
- peer.sh
  - Script to launch an instance of a peer with some parameters.

Usage: peer.sh <peer\_num> Eg. sh peer.sh 1.0 1
- Others scripts
  - Action scripts (backup.sh, restore.sh, delete.sh) run in peer 1 with default file as image1.png
  - The script 'reclaim.sh' runs in peer 2 with a size of 140000 (reclaiming all used memory above 140KB).
  - And the script state.sh retrieve information about peer 1 and peer 2. - All scripts use the **localhost** ip by default (127.0.0.1).