User Manual of Ganger

Zhihui Jiao

April 10, 2012

Version: 1.0.0

Contents

1 Introduction

1.1 What is Ganger

Ganger is a simple automate tool. It can automate deploy packages, launch target processes, update packages deployment(kill the old process and launch the new one), restart target processes and monitor the status of target process.

It saves your time for development, and make it easier to deploy your systems on many target machines simultaneously.

1.2 Terminology

There are some terms you should know when using `ganger`:

- **project**: project defined as a configuration file which describes the packages belong to it, the target machines these packages should deploy to, the startup command used to launch target process etc.
- **package**: compressed binary executables which contains the target program.
- **process**: process runs on the target machine, each process will belongs to a package.

1.3 Features

- ***automatation***: help you do some automate tasks, for example: deploy packages, start processes, stop processes and update deployments etc.

  Note: deploy action **ONLY** support compressed binary executables(.tar.gz) currently.
- ***lifecycle management***: you can start, stop, restart or update the target process.
- ***dependence management***: we organize the package as a directed graph with no cycles, when a process of a package crashed or started, the processes of the package which depend on it will be notified.

  Automate action will operate on packages as the topological order of the dependence graph(or reverse order for actions similar to `stop`).
- ***status monitor***: monitor the target process's status.

2 Key Components

2.1 Master

The master manage the whole system, accept the configurations of user, and do the work based on the configuration: deploy packages, launch process etc. And there is a simple console for user.

2.2 Slave

The slave accept the command send from master, and execute the command.

2.3 ZooKeeper

ZooKeeper(`http://http://zookeeper.apache.org/`) is used to maintain the state in a central location and is used for its powerful notification capabilities (ephemeral nodes and watchers).

3 Dependences

3.1 SSH

When deploying packages, slaves need an ssh access to the remote master node without password. Slaves will pull(rsync) packages from the package root in master node to a default local path, and extract them.

3.2 ZooKeeper

You need set up a ZooKeeper instance, and start ganger with the ZooKeeper address.

4 Compile From Source

You can get compressed binary executables by running the `release` target of ant. It will produces two compressed files: one for master and the other for slave. These two files resident in `target\release` after you run the `release` command.

5 How Does It Work

1. You define a project configuration file and load it through command of master(see Example 7)).
2. You tell ganger to execute the deploy task(or start, stop etc.).
3. Master read the configuration info from project configuration, send the task command and package info to the slave.
4. Slave execute the command with the command info and give a feedback to master.
5. Master receive the feedback and consider the task has done.

6 Command List

You can get a full list from the master console by typing help.

```
    >./ganger zkAddress
                zkAddress is the ZooKeeper address
    ganger > load project
                load project from configuration file
    ganger > deploy project [packages...]
                deploy packages defined in the project
    ganger > launch project [packages...]
                launch process defined in the project
    ganger > update project [packages...]
                update deployed packages and restart the process
    ganger > stop project [packages...]
                stop target process
    ganger > restart project [packages...]
                restart target process
```

7 Example

Configuration file A sample configuration(as a Java property file, locate in `sample/xmpp.properties`):

```
project.name=xmpp
project.packages=master,openfire,proxy,robot
project.packageRoot=/home/space/automate/xmpp

# master
package.master.version=1.0.0
package.master.fileName=master-1.0.0.tar.gz
package.master.deployPath=/home/space/automate/deploy
package.master.launchCmd=bin/mxmaster.sh
package.master.targets=10.100.82.217

# openfire
package.openfire.version=3.7.2
package.openfire.fileName=openfire-3.7.2.tar.gz
package.openfire.deployPath=/home/space/automate/deploy
package.openfire.launchCmd=bin/openfire.sh
package.openfire.targets=10.100.82.216,10.100.82.215
package.openfire.deps=master

# proxy
package.proxy.version=1.0.0
package.proxy.fileName=proxy-1.0.0.tar.gz
package.proxy.deployPath=/home/space/automate/deploy
package.proxy.launchCmd=bin/wxproxy.sh
package.proxy.targets=10.100.82.215
package.proxy.deps=openfire

# robot
package.robot.version=1.0.0
package.robot.fileName=robot-1.0.0.tar.gz
package.robot.deployPath=/home/space/automate/deploy
package.robot.launchCmd=bin/mxrobot.sh
package.robot.targets=10.100.82.214
package.robot.deps=openfire
```

Config Fields

```
project.name
        # project name
project.packages
        # packages belong to this project
project.packageRoot
        # package root of this project(contains the initial packages)

package.PKG.version
        # package version
package.PKG.fileName
        # full package name
package.PKG.deployPath
        # deploy path on target machine
package.PKG.launchCmd
        # command for launching target process
package.PKG.targets
        # the address(ip) of target machine
package.PKG.deps
        # package that this package dependent on
```

Usage You can use following commands to load a project configuration, deploy the packages and launch the processes on the target machines:

```
>./ganger 10.100.82.218
ganger > load xmpp.properties
ganger > deploy xmpp
ganger > launch xmpp
```

and then, you can use status to check system status:

```
ganger > status xmpp
```

this command will show the slave status and process status.