Table of Contents

Creating the release

Note: You may find it useful to start a gnu `screen` session under the gangage account. You can then create new screens and log into the gangalb and gangaat accounts in each of these. Usually 4 screens is good. One to run the release manager tool, and 3 for the tests on the gangage, gangalb and gangaat accounts respectively. The account has a `.screenrc` file which specifies `bash` as the default shell and `Ctrl-p` as the screen command sequence. Remember that if you press "Ctrl-P" followed by "esc" you can use the arrow keys to scroll up and down. Use "return" a few times to get out of the scroll mode.

Note: Use "lxplus5" in place of "lxplus" until we have confirmation that the lxplus6 service works without any major issues.

Prerequisites

When you take over the release management you should do these steps:

- configure submission to the Grid
  - Copy your userkey.pem and usercert.pem files into the .globus directory of the gangage account
  - Generate a long-life proxy in the normal way for your VO, and copy it to `/afs/cern.ch/user/g/gangage/private/ganga.proxy`
    - `source /afs/cern.ch/project/gd/LCG-share/current/etc/profile.d/grid_env.sh`
    - `voms-proxy-init -voms [your_VO in all lowercase]` -vomslife=168:00
    - You can find your newly created proxy in /tmp/x509up_u[UID] `cp /tmp/x509up_u`id -u` /afs/cern.ch/user/g/gangage/private/ganga.proxy`
  - On the gangage account, adjust .gangarc VO name (in two places).
  - quick test if submission works: Job(backend=LCG()).submit()
- communication
  - make sure that you are subscribed to relevant mailing lists so you can send email announcements about the releases: http://cern.ch/ganga/mail
  - update the file /afs/cern.ch/sw/ganga/www/developers/tags/release_manager.php to have your email address in it as release manager. Check that the tags page https://ganga.web.cern.ch/ganga/developers/tags/ still works and that after selecting one of the releases you should see your name at the bottom of the page. When a tag is submitted you will now receive an email.

Make the release

- Get proxies for test accounts. They are valid for 168 hours so you may be able to fit more than one testing cycle in without having to request them again.
  - Get an Atlas person to run the command `/afs/cern.ch/user/g/gangage/public/generate_atlas_proxy.sh` **Make sure the correct proxy in ~gangaat/private is updated**
  - Get an LHCb person to run the command `/afs/cern.ch/user/g/gangage/public/generate_lhcb_proxy.sh`
- login into the generic release account (**gangage**) on lxplus
- start the release-tool with the release version as a parameter, e.g. `~/ganga/release/tools/ganga-svn-release 6.0.14` The release procedure consists of 11 steps split in two phases : first we create the pre-release version (steps 1-6) used internally for testing and then the final release (7-11).

Short explanation of the release steps:

Create the pre-release

Note: You can recreate the pre-release version (steps 1-6) as many times it's necessary

1 - COLLECT:

Collect tags and release notes from the tags page and writes them into `~/ganga/release/config/PACKAGES-VER` file and `~/ganga/release/ReleaseNotes-VER`.

2 - PRE:

Creates the pre-release version into `/afs/cern.ch/sw/ganga/install/5.X.Y-pre` directory using the tags collected in step 1. At the end of this step there is a question asking whether the release tool should notify the developers by email that the prerelease is ready.

3 - GENERATE CONFIG FILES

Run the commands printed by this step for each of the accounts (gangage, gangaat and gangalb). For example:

```
Enter your choice:3
doing: cd /afs/cern.ch/sw/ganga/install/5.7.6-pre; svn update templates

At revision 6967.
login to gangage@lxplus and run this command:
sh ~/test/generate_generic_config_files.sh 5 7 6

login to gangalb@lxplus and run this command:
sh ~/test/generate_lhcb_config_files.sh 5 7 6

login to gangaat@lxplus and run this command:
sh ~/test/generate_atlas_config_files.sh 5 7 6

When you have executed the commands for all accounts, press Enter to continue ...
```

At the moment only the command on the gangalb account are required. The other commands can be skipped.

4 - COMMIT CONFIG FILES

This step can now be skipped

5 - TEST:

Be aware that you can only run one instance of the testing framework per user account (ie `gangaat` , `gangalb`, `gangage` etc) at any given time.

Running the testing framework is an important part of the release procedure. Execute step 5 and it will show you which command to run on each of the user account.

- Make sure you have afs tokens and Grid proxies valid for long enough to finalise the testing framework (can be more than 12 hours). If you believe that your afs token will expire while tests are still running, type `Ctrl-z` to halt the tests, then `kinit` to renew the token, and `fg` to resume the tests.
- Due to lack of verbosity, some tests might appear as hanging. They are not. A dot (.) in the standard output is a successful test, a F is a failed test. The detailed reasons of failures appear when the entire set of tests is over.

- When all tests are finished the release manager will receive a notification email so he can proceed then with step 6 of the release tool.

6 - TEST REPORTS:

This step will create the web pages with the result of the tests. If tests are available from remote sites they will be downloaded first. Failing to download tests from other sites should not prevent a release from being created.

- You can safely ignore warning like `WARNING:figleaf.htmlizer:CANNOT OPEN: TestLHCbDataset.py` when creating the web pages for the test reports:

At the end of this step there is a question asking whether the release tool should notify the developers by email that the test reports are ready.

7-11 Create the release

Note: You should run the following steps (7-11) only once for a given release

- 7 - TAG: make a GLOBAL CVS TAG using the a pre-release in the prereldir
- 8 - SETVER: Sets the release version in the $Name fields. This was previously automatic with CVS, but for now needs this extra step. Pay attention to the prompts... you need to verify that the files are patched correctly.
- 9 - RELEASE: make a release based on a previously created GLOBAL TAG and create the distribution kits. At the end of this step there is a question asking whether the release tool should notify the developers and ganga-project group by email that there is a new release ready. If you are not an ATLAS (LHCb) user, ask one of the ATLAS (LHCb) developers to resend the email for the release to the ATLAS (LHCb).
- 10 - GENERATE LHCB MANUAL - follow the instructions from the release tool - login to gangalb and execute the given command
- 11 - PREPARE NEXT development cycle: prepare tags dir for the next releases

Release tool maintenance

Tags Page

Developers' tags are collected using online form available at :
https://ganga.web.cern.ch/ganga/developers/tags/ Internally the form is using simple text-files to save the tags information kept in `TAGS_DIR=/afs/cern.ch/sw/ganga/www/developers/tags/`.
For each release there is a `$TAGS_DIR/data/release_version` directory containing the tags files.
Each tag has its own file storing the info for a given release. E.g:
`$TAGS_DIR/data/5.0.0-beta4/ganga.python.Ganga.Lib.LCG` stores the information for `ganga/python/Ganga/Lib/LCG` (sub)module to be used in 5.0.0-beta4 release.
During the release `ganga-release` tool is using these files in two different steps:

- `step 1 - COLLECT: collect tags from the tagsdir into PACKAGES file and create the new release notes file`
- `step 11 - PREPARE NEXT development cycle: prepare tags dir for the next releases` During this step a new directory is created in `TAGS_DIR/data` and all the tags files for the current release are copied in it. (removing the release notes so the tags page will display the **last** version of each module for the new release)

Adding a new module in the tags form

To add a new Ganga package(module) in the tags form you need to create a new file in `TAGS_DIR/data/release_version/` dir named according to the above convention (i.e: `ganga.python.GangaPACKAGE.dir.subdir`) The format for the tag file is the following:

```
Line1  :  maintainer(s) name
Line2  :  Tag version
Line3- :  Release
                Information
```

Opened Ganga releases

During the development there might be multiple Ganga versions available on Tags page. These release versions are defined in `TAGS_DIR/open_releases` one per line. In principle you don't need to edit this file by hand (`ganga-release` tool is managing this file for you) but there might cases when you want to forcibly close a release. In this case you need to manually remove `TAGS_DATA/dir/ganga_release` dir and remove `ganga_release` line from `TAGS_DIR/open_releases`

Authorize a new developer to use the tags form

The list of authorized developers that can submit tags to our tags page is maintained in `TAGS_DIR/.htaccess`. Should be straight-forward to add/remove new developers.

LHCb specific information

Instructions for how to create an LHCb specific release and testing it is given at HowToReleaseProcedureLHCb. This is only required for debugging.

Creating a Beta Release using Unmerged Branches

If there are significant developments in a branch that require extensive testing in a release, but aren't ready to be merged to the trunk yet, the best idea is to create a beta release:

1) Run the release tool with the version of Ganga that the branch is based on and open for tags the release (e.g. 5.4.0-beta)

2) Go into `/afs/cern.ch/sw/ganga/www/developers/tags/data/` and in the directory corresponding to the beta release will be the tags

3) Replace the tags in this directory with any associated with the branch to be tested so that these are used instead. As an example, for the lazy loading branch, the tag `Ganga-Base-XMLMigration-1-2` contained the whole of Ganga Core. The following tags were therefore deleted:

```
/afs/cern.ch/sw/ganga/www/developers/tags/data/5.3.6/ganga.python.Ganga
/afs/cern.ch/sw/ganga/www/developers/tags/data/5.3.6/ganga.python.Ganga.GPIDev.Credentials
/afs/cern.ch/sw/ganga/www/developers/tags/data/5.3.6/ganga.python.Ganga.GPIDev.Lib.GangaList
/afs/cern.ch/sw/ganga/www/developers/tags/data/5.3.6/ganga.python.Ganga.GPIDev.Lib.JobTree
/afs/cern.ch/sw/ganga/www/developers/tags/data/5.3.6/ganga.python.Ganga.GPIDev.Persistency
/afs/cern.ch/sw/ganga/www/developers/tags/data/5.3.6/ganga.python.Ganga.Lib.Condor
/afs/cern.ch/sw/ganga/www/developers/tags/data/5.3.6/ganga.python.Ganga.Lib.Interactive
/afs/cern.ch/sw/ganga/www/developers/tags/data/5.3.6/ganga.python.Ganga.Lib.LCG
/afs/cern.ch/sw/ganga/www/developers/tags/data/5.3.6/ganga.python.Ganga.Lib.Mergers
/afs/cern.ch/sw/ganga/www/developers/tags/data/5.3.6/ganga.python.Ganga.Lib.MonitoringServices.AR
/afs/cern.ch/sw/ganga/www/developers/tags/data/5.3.6/ganga.python.Ganga.Lib.MonitoringServices.MS
/afs/cern.ch/sw/ganga/www/developers/tags/data/5.3.6/ganga.python.Ganga.Lib.MonitoringServices.Oc
/afs/cern.ch/sw/ganga/www/developers/tags/data/5.3.6/ganga.python.Ganga.Lib.Remote
/afs/cern.ch/sw/ganga/www/developers/tags/data/5.3.6/ganga.python.Ganga.Lib.Root
/afs/cern.ch/sw/ganga/www/developers/tags/data/5.3.6/ganga.python.Ganga.scripts
```

and replaced by:

```
/afs/cern.ch/sw/ganga/www/developers/tags/data/5.4.0-beta/ganga.python.Ganga
```

that contained:

```
kuba
Ganga-Base-XMLMigration-1-2
```

4) Re-run the release tool with the beta release as the argument and go through the prcoess of creating the pre-release. Note that it should be checked that the appropriate branches have been included!

Hotfix procedure for ATLAS

These instructions are for making very small yet urgent fixes to ganga. These steps assume no changes to the config templates are necessary (which should be the case for hotfixes).

1. Start the release tool with the version number of the release to be fixed. E.g. to release 5.5.3-hotfix1, start with 5.5.3

2. Use the prepare next command to open the hotfix release, e.g. 5.5.3-hotfix1

3. Go to the tags page and submit the new tag. (Remember to also submit for the real next release too).

4. Start the release tool with the hotfix name, e.g. 5.5.3-hotfix1

5. COLLECT tags and make the PRE.

6. Test the pre separately. Make sure that everything is working. (We will not run the full testing procedure).

7. Start the release tool with the hotfix name.

8. Make the release with TAG, SETVER, RELEASE, PREPARE NEXT

- -- UlrikEgede - 18-Jul-2013

This topic: ArdaGrid > HowToReleaseProcedure
Topic revision: r63 - 2014-05-07 - MattWilliams