Table of Contents

Ganga LHCh FAQ............................................................................................................................................1
  Installing Ganga........................................................................................................................................1
    How do I install Ganga?.........................................................................................................................1
  Starting Ganga........................................................................................................................................1
    Problems with Ganga startup............................................................................................................1
    How do I start a different version of Ganga?.....................................................................................1
    Can I get help inside Ganga?.............................................................................................................2
    Where are the release notes?..............................................................................................................2
    How can I change the verbosity of Ganga?.........................................................................................2
    How do I turn off the job monitoring loop in Ganga?.......................................................................2
  Known Ganga Issues.............................................................................................................................2
    How can I ban Grid sites for a Ganga job?.........................................................................................4
    What should I do if my DIRAC job is stuck in "completing"?.........................................................5
    What should I do if my DIRAC job is stuck in "submitting"?............................................................5
    How can I use DIRAC API commands for a job created by Ganga?..............................................4
    How can I install Grid sites for a Ganga job?....................................................................................4
    How can I set which Grid site my Ganga job should run at?............................................................4
    What should I do if my DIRAC job is stuck in "submitting"?............................................................5
    What should I do if my DIRAC job is stuck in "completing"?............................................................5
    How can I restart the Ganga monitoring loop to force a status update for a given job or subjob?.................................................................................................................................5
  Jobs finished but still in submitted state (monitoring is not running).............................................5
  My jobs are shown as Running when they are in fact Stalled, how can I see the exact DIRAC backend status in Ganga?........................................................................................................6
  Which version of DIRAC does Ganga use?.........................................................................................6
  How do I get information about my grid proxy inside of Ganga?....................................................6
  How do I renew my grid proxy inside Ganga?.....................................................................................6
  How do I run an executable job that uses input files on the Grid as arguments to the script?........6
  How can I download output data only from a subset of DIRAC jobs that were resubmitted?.........7
  Ganga and Gaudi jobs..........................................................................................................................8
    How do I know if a Gaudi job processed all the input files, or just skipped some of them (Oct 18 2010)..............................................................................................................................8
    I get an error like ERROR Database not up-to-date. Latest known update is at 1273462317.0, event time is 12734 (May 14 2010)........................................................................................................8
    I get an error like coral::LFCReplicaService::LFCException: LFC host has not been defined (Oct 5 2010).......................................................................................................................9
    How do I run jobs that require a grid proxy on a LSF?................................................................9
    Can I submit jobs to LSF if I'm not running on lxplus?................................................................9
    How do I debug subjobs associated with a particular job?.............................................................9
    Can I submit Ganga jobs from the LSF batch system?................................................................10
    How to avoid the CPU time limit on lxplus machines?..................................................................10
    Correct platform setting, or, "What do I do if some of my DIRAC jobs fail with "Could not retrieve Tool"? (28 June 2019).............................................................................................................10
  Ganga Setup/Configuration..................................................................................................................11
    How do I define and load utility functions from within ganga? (16 Nov 2009)..........................11
    How to use Ganga functionality from a Python script................................................................11
    How do I use Root within Ganga?..................................................................................................11
    Root merging fails, segfaults recursively or takes an infinite amount of time or memory or writes a file of infinite size to disk......................................................................................12
    How do I change where my gangadir is?.......................................................................................12
  How can I run the monitoring loop for a subset of jobs?...............................................................12
  How do I set the Ganga output location for a cluster outside of CERN?.......................................12
Table of Contents

Ganga LHCh FAQ

Data Sets and Data Files......................................................................................................................12
   How do I make a dataset that only contains the files added since my last run?..........................12
   How do I make a dataset only containing data taken between specific runs?..........................13
   How do I use the outputdata from one Grid job as the inputdata for another?.........................13
   How do I make a new job that runs only on the inputdata from some failed subjobs?............14
   How do I extract an LHChDataset from a Gaudi-style options file?........................................14
   Why am I getting a "No ancestors found" error?......................................................................14
   How do I deal with replicas (e.g. replicate, get replicas, remove replicas) in Ganga?...............14
   How do I download a file from Grid storage?..........................................................................14
   How do I remove a file from Grid storage?.............................................................................15
   How do I remove all files from a Dirac job?...........................................................................15
   How do I upload a file to Grid storage?..................................................................................15
   How can I put a file bigger than 10 MB in my input sandbox for a Dirac job?.........................15
   How do I run on SDSTs?......................................................................................................15
   Streaming DST from different SE (CERN)..............................................................................16
   How to run over locally-stored DST?.....................................................................................16
   How to select LHCh dataset using Bookkeeping information (for example TCK)?....................16
   What are the pros and cons of MassStorageFile and DiracFile?............................................16

Retrieving/merging output...................................................................................................................17
   How do I turn off/on default downloading of the output sandbox for failed jobs?...............17
   How do I download an output sandbox by hand?.....................................................................17
   How do I control output file locations? (3 Dec 2008).............................................................17
   How do I retrieve outputdata (large files)? (27 Jan 09)..........................................................18
   Do the mergers only work for local files? (4 Jun 08).................................................................18
   Can I prevent stdout from being in my output sandbox/Can I put stdout in output data?.....19

Bookkeeping........................................................................................................................................19
   How do I get an LHChDataset from the bookkeeping GUI in Ganga?..................................19
   How do I get a "path" for the BKQuery object?...............................................................19
   How can I find out the tags / DDDB tags / project versions / options for a given BK selection?.................................................................................................................................19
Ganga LHCb FAQ

This page will be used to answer questions that frequently arise on the lhcb-distributed-analysis mailing list. Everybody is welcome to update and correct the page.

Installing Ganga

How do I install Ganga?

If you're working on a machine where LHCb software is available through the cvmfs file system or if working on an lxplus machine at CERN, then you don't have to install anything. If you want to install Ganga and use it outside the LHCb environment, simply do `pip install ganga` which will install all the dependencies as well. See the `pip` manual page if you have questions about how to install in your userspace.

Starting Ganga

Problems with Ganga startup

Ganga may potentially appear to stall due to a few different reasons, some of these are:

1) Stale lockfiles from previous ganga sessions which did not close correctly.

Solution: remove all files in `~/gangadir/repository/$USER/LocalXML/6.0/sessions` then restart ganga.

2) First time launching Ganga >6.1.x

This version of ganga introduces a better caching system which will require reading all data from disk once on startup, but future startups will be much faster.

3) Corrupt repo/repo-cache

If this occurs ganga will read all data from disk again as in step 2 for data security and then future startups should be much faster

4) AFS issues

Ganga may fail to start correctly if your afs partition (userspace) is full or has a slow connection. If possible try from another machine or clear some disk space and try again

5) Other issues

If you're waiting for more than 2-3min then try releauching with `ganga --debug`. If it isn't obvious what is wrong at this stage then ask for help on the mailing list attaching the output of what you got here. (Apologies it is a little verbose)

How do I start a different version of Ganga?

If you want to use a version of Ganga other than the default one on Lxplus, do, e.g.,

```
ganga --ganga-version 7.1.14
```

or
Bear in mind though, you should always use the latest version of Ganga!

**Can I get help inside Ganga?**

Yes. For any class and/or method, simply type `help(ClassName.methodName)`.

**Where are the release notes?**

The release notes can be found on the Github release page.[1](#)

**How can I change the verbosity of Ganga?**

You can change the following in your $HOME/.gangarc

```
[Logging]
# top-level logger
Ganga = DEBUG
# LHCb plugin verbosity
GangaLHCb=DEBUG
```

**How do I turn off the job monitoring loop in Ganga?**

You can start ganga via:

```
ganga -o '[PollThread]autostart=False'
```

or add the same option to your $/.gangarc file directly, or use

```
ganga --no-mon
```

**Known Ganga Issues**

**Ganga and Dirac**

**Where is the Dirac Monitoring Page?**

[HERE](#) is the current DIRAC monitoring page. (lhcb-portal-dirac.cern.ch/DIRAC)

There is also a mirror site at CERN which can be accessed [here](#).

**Why has my job been marked as failed in Ganga and completed in DIRAC?**

First perform some of the usual checks.

- Do you have enough disk space for 3 x sandbox size to be downloaded to your gangadir? Remember Ganga will download the output from 3 subjobs in parallel and if you're placing your files on MassStorage these *must* be stored in your gangadir before being uploaded to MassStorage.
- Do you have an active kerberos token if on AFS? If in doubt try running kinit -R from your terminal
- Are you certain the job was completed in DIRAC?

How do I start a different version of Ganga?

---

[1](#): Bear in mind though, you should always use the latest version of Ganga!
• Is the job currently downloading the output but is taking a while? If you see something like 'Finalizing job x.y' in the output of the command queues wait for it to complete!

If your job has been marked as completing in ganga and is not currently being completed in ganga (i.e. is not in the queues summary) you will need to first mark the job as failed. (This will only happen if ganga was closed in the process of downloading the output from a job)

subjob.force_status('failed')

If your job has been marked as failed and you don't have the expected output sandbox in your output workspace for the job try marking the job as newly submitted again in Ganga and let it try again.

subjob.backend.reset()

Ganga 6.1.12 will make 5 attempts to finish completing a job before it will abandon the job and mark it failed.

**What can I do if I see "Not authorized in DIRAC"?**

The "Not authorized in DIRAC" error message can come about due to several reasons such as:

• Incorrectly configured installation of DIRAC
• Expiry of your VO membership
• A change rather than renewal of your certificate (DN)
• Incorrect registration in the LHCb Virtual Organization (VO)
• ...

Hopefully the following checklist should reduce the list of possibilities:

• Check your LHCb VO membership at the LCG VOMS page, select "Member Info" then "VO Exp Date" and search
• Check that your certificate is valid and was correctly converted for Grid usage as in the Grid certificate FAQ
• When using a local Ganga installation check that the steps described in the LHCb Ganga installation instructions were followed
• In case of an error like:

  Can't create a proxy: Can't contact DIRAC CS: Reason(s):
  Can't connect to dips://lhcb-conf-dirac.cern.ch:9135/Configuration/Server:

  • Check that you can connect to the port number in the error message to eliminate firewall issues e.g.

    (stuart@lhcb)$ telnet lhcbprod.pic.es 9135
    Trying 193.109.175.160...
    Connected to lhcbprod.pic.es.
    Escape character is '^]'.

    • If this is unsuccessful (i.e. you don’t see "Connected") you should contact your sys admin to open the affected port.

In case the above does not resolve the situation please email the distributed analysis list with the full output of lhcb-proxy-info and the context in which the error appears e.g. local or lxplus Ganga installation, any recent
changes in your certificate.

**Why do all my jobs fail with "Failed to setup proxy: Error retrieving proxy"?**

There are several causes of the above failure but the most common ones are:

- A change or renewal of your certificate (DN)
- Recent renewal of the LHCb Virtual Organization (VO) membership
- Instability of the VOMS servers that are used to extend your proxy - this affects everyone in the same way

For example, if jobs were submitted with a CPU time of several days and your certificate is due to expire within that period DIRAC will unsuccessfully try to obtain a proxy long enough to satisfy this job requirement. Therefore DIRAC sees an error in retrieving your proxy and fails the job. As soon as the renewal / change of certificate procedure is completed and you have submitted a new job with your new credential then this problem will disappear (at least for another year 😊)

**How can I use DIRAC API commands for a job created by Ganga?**

Anything added to the `diracOpts` attribute of a `Dirac` for a job will be added to the Python submission file for your DIRAC job.

The `LHCbJob` is called `j`, so, e.g., you could override Ganga's naming scheme by doing `j.backend.diracOpts = j.setName("SomeName").`

**How can I ban Grid sites for a Ganga job?**

This is best done through the settings attribute of the Dirac backend. These can be changed at any time and are used during submit and resubmit. The setting to ban a list of sites is:

`j.backend.settings['BannedSites'] = ['LCG.Some.Site','LCG.SomeOther.Site'].`

(This can also be done using the `diracOpts` attribute, `j.backend.diracOpts = j.setBannedSites(['Site1','Site2'])` however, this attribute cannot be changed for resubmit.)

**How can I set which Grid site my Ganga job should run at?**

This is performed through the `Dirac.settings` attribute. These can be changed at any time and are used during submit and resubmit.

The setting to force the job to go to a certain site is: `j.backend.settings['Destination'] = 'LCG.Some.Site'.`

Remember the site is in the same notation as used elsewhere in Dirac.

(This can also be done using the `diracOpts` attribute `j.backend.diracOpts = j.setDestination("Site")` however, this attribute cannot be changed for resubmit.)

What can I do if I see "Not authorized in DIRAC"?
What should I do if my DIRAC job is stuck in "submitting"?

`j.kill()`, `j.submit()` and `j.resubmit()` will not work on a job that is stuck in the submitting state.

Instead you should use `j.force_status("failed")` to force the job into the failed state then use `j.resubmit()` as normal.

For a job which got stuck submitting multiple subjobs you may wish to try:

```python
j=jobs(PrimaryJobId)
for sj in j.subjobs.select(status="submitting"):
    sj.force_status('failed')
    sj.resubmit()
for sj in j.subjobs.select(status='new'):
    sj.submit()
```

What should I do if my DIRAC job is stuck in "completing"?

If your DIRAC job is stuck in "completing", then this means that something probably went wrong while attempting to download your output. The simplest thing to try is to do `job.backend.reset()` and see if it works on a 2nd attempt (which would be true if the problem was something intermittent). If it fails again, please take note of any warnings and/or error messages and email the lhcb-distributed-analysis mailing list with the job details.

Note: It's also possible that you've run out of disk (or AFS) space. Please check this prior to emailing the list. This is common if you set up your job incorrectly such that it produces a very large output sandbox (typical AFS quota is 500 MB, so if you're sandboxes are ~100 MB or more, you're in trouble).

How can I restart the Ganga monitoring loop to force a status update for a given job or subjob?

For Dirac jobs the backend has a reset method which puts the status back to submitted and then the monitoring is redone (including downloading output). As for jobs stuck in "completing" the recipe is to apply the `job.backend.reset()` method. For a job with subjobs:

```python
j=jobs(PrimaryJobId)
for sj in j.subjobs.select(status="failed"):
    sj.backend.reset()
```

Jobs finished but still in submitted state (monitoring is not running)

The main causes and solutions for jobs in a completed state but still "submitted" in Ganga are:

- AFS token being expired, for which you can try `reactivate()` to restart the monitoring loop
- DIRAC monitoring is turned off because your Grid proxy is expired, the following will ensure a proxy is created: `gridProxy.create()` or `gridProxy.renew()`
- Jobs were rescheduled in the DIRAC page and monitoring needs to be reset in Ganga via `job.backend.reset()`

Here is another thing you can try, if this does not work. Start ganga without the monitoring on.

```
> ganga --no-mon
```
Then, start the monitoring for a specific job you want to try to re-activate the monitoring (here jobID = 291 as an example), then check with the "queues" command:

```python
ganga[ ]: runMonitoring(jobs.select(291))
ganga[ ]: queues
```

You may need to do this several times, hopefully you start to see jobs in the "Finaliing" stage with the "queues" command. Good luck!

**My jobs are shown as Running when they are in fact Stalled, how can I see the exact DIRAC backend status in Ganga?**

You can look at the `backend` object for the subjobs. In particular `backend.id`, `backend.status` and `backend.actualCE` will be of interest. The same information is also available from the DIRAC job monitoring page. Use the `backend.id` to match up the information.

**Which version of DIRAC does Ganga use?**

The version is reported at start-up on `stdout`. By default it is selected as the latest stable version but the behaviour can be changed in the `.gangarc` file.

**How do I get information about my grid proxy inside of Ganga?**

This is available from the `credential_store` object. See `help(credential_store)` for info.

**How do I renew my grid proxy inside Ganga?**

You can interact with all proxies through `credential_store.credential_store[DiracProxy()].renew()` will renew your grid proxy, but only if it’s invalid or expiring soon. `credential_store.renew()` renews all credentials that are invalid or will expire soon.

If you are running a long process, you may want to create a new proxy instead. You can do so with `credential_store.create(DiracProxy())`, ensuring that you’ve maximized the valid time of your proxy.

See also `help(credential_store)`.

**How do I run an executable job that uses input files on the Grid as arguments to the script?**

In order to pass the input data of your job as an argument to an arbitrary executable we use inputfiles, DiracFile files are automatically downloaded in the local directory for most backends. Note that files should not have the same name for this to work. The following settings should be made to get this to work:

- Put the LFNs into the `j.inputdata` this will make DIRAC match the job to a site that has the requested input data. This is made available to gaudi applications as data.py.
- Add LocalFiles and MassStorageFiles to the `j.inputfiles`. These are sent to the worker node as part of the Dirac inputsandbox.
- Add the LFNs you’ve uploaded to the grid as `j.inputfiles`. These files will now sit as local files on the worker node when your job starts running.
- Set the arguments of the Executable `j.application.args` to the list of input data for the job
- Note that merged files are normally around 5GB in size, therefore one input file per job would be optimal.
import os

# Running an executable script with input data as arguments
yourScript = 'printArgs.py'

# Create the job object specifying the executable as usual
j = Job(application=Executable(), backend=Dirac())
j.application.exe = File(yourScript)
j.backend.settings['CPUTime'] = 500

yourData = ['LFN:/lhcb/MC/MC09/DST/00005138/0000/00005138_00000232_1.dst']
data = LHCbDataset(yourData)

# Set the input data for the job as usual
j.inputdata = data

# Now get the input data files as a python list, starting # from the LHCbDataset object. Note that above we # could just use "yourData" but this way is cleaner:
inputDataList = []
for i in data.files:
    inputDataList.append(i)

# Now set the script arguments to the list of input data # this can either be the full LFNs or (since we know the # files will be on the local disk, just the file name
inputDataList = [i.namePattern for i in inputDataList]
j.application.args = inputDataList

# job.inputfiles handles both LocalFile and DiracFile objects # all thats required to tell a job about a file is to load it # here
j.inputfiles = inputDataList

# print j
j.submit()

where the executable script printArgs.py is simply:

#!/usr/bin/env python
import sys

print 'This is a simple python script that prints "sys.argv", the list of arguments:
%s' % (sys.argv)

The output of testing the above job would be:

(stuart@lxplus)$ cat ../..\gangadir\workspace\stuart\LocalXML/13/output/Script1_Ganga_Executable.log
<<<<<<<<<< exe-script.py Standard Output >>>>>>>>>>
This is a simple python script that prints "sys.argv", the list of arguments:
['/scratch/lhcb04920970.ccwl9078/tmp/home_crm02_556854492/CREAM556854492/10829047/printArgs.py',

How can I download output data only from a subset of DIRAC jobs that were resubmitted?

The easiest thing to do would be something like:

import os

for sj in j.subjobs:

How do I run an executable job that uses input files on the Grid as arguments to the script?
if not os.path.exists(sj.outputdir + '/some.file'):
    for f in sj.outputfiles.get(DiracFile):
        f.localDir = sj.outputdir
        f.get()

where I assume you know the name of one of the DiracFiles files and that any job that has downloaded that
file has downloaded the entire contents.

The call j.backend.getOutputDataLFNs() works in Ganga regardless of whether the job still exists in
DIRAC, the LFNs are cached in the DiracFile objects when Ganga retrieves the Dirac sandbox. Therefore if
the above fails (or you do not know the name of the file) it is also possible to search on the local disk for the
LFNs determined from the j.backend.getOutputDataLFNs() call and then download them via df.get() to
download to your home directory or df.get('/path/to/download/to') where df = DiracFile
(lfn='/some/lfn.file') e.g. see "How do I download a file from Grid storage?" below.

Ganga and Gaudi jobs

How do I know if a Gaudi job processed all the input files, or just
skipped some of them (Oct 18 2010)

In the Gaudi framework, if a job finds that it can't open an input file, it will merely continue with the other
files in the input data. This is efficient, as a single failure does not cause a whole job to abort. On the other
hand it means that jobs that only partially process there input are labelled in Ganga as "completed". To extract
this information, you need to add the XML summary to your jobs. See XMLSummaryFAQ for the details of the
implementation. Once this is written into your Ganga job, you can now process and merge this information
from within Ganga. See the reference manual for the GaudiXMLSummary and GaudiXMLSummaryMerger
classes.

I get an error like ERROR Database not up-to-date. Latest known update is
at 1273462317.0, event time is 12734 (May 14 2010)

You are running on data taken very recently from the detector where the conditions have not yet been entered
into an SQLDDDB snapshot.

If running outside of Ganga the following should be used to set your environment:

> SetupProject <Project> <Version> --use-grid
> lhcb-proxy-init

and the following options should be specified in your job:

# Additional options for accessing Oracle conditions DB.
from Configurables import CondDB
CondDB(UseOracle = True)

In addition add the option below if running with the DIRAC backend.

importOptions("$APPCONFIGOPTS/DisableLFC.py")

Another reason for this error is that you are running locally outside CERN and have failed to update your
local version of the SQLDDDB.

Do not run with the useOracle = True option unless required as it slows job your jobs and cause
trouble for others.
I get an error like coral::LFCReplicaService::LFCException: LFC host has not been defined (Oct 5 2010)

You are using the Oracle database in a local job, but have no valid connection

If running outside of Ganga the following should be used to set your environment:

> SetupProject <Project> <Version> --use-grid
> lhcb-proxy-init

Then set

# Additional options for accessing Oracle conditions DB.
from Configurables import CondDB
CondDB(UseOracle = False)

And DO NOT INCLUDE DisableLFC.py

If this produces a different error because your job does require oracle, consider running on a different file, or running on the grid following above FAQ

*Do not run with the useOracle = True option unless required as it slows job your jobs and cause trouble for others.

If you absolutely have to run locally, and have to use this file, then you should add '--use-grid' in your job.application.setupProjectOptions, and follow above FAQ

You will need a valid grid proxy, which may mean following FAQ below

How do I run jobs that require a grid proxy on a LSF?

You need to setup your environment so that your grid proxy gets placed in your home area on AFS (instead of on the local machine's /tmp area).

To do this, simply set X509_USER_PROXY = $HOME/somewhere. Then, whenever you do lhcb-proxy-init your proxy will be placed there (which is visible from any LSF worker node).

Can I submit jobs to LSF if I'm not running on lxplus?

Yes, using the Remote backend. See help(Remote) for details. But why would you want to?

How do I debug subjobs associated with a particular job?

Lets say subjob 8 of job 25 is the one you want to look at; do the following:

# Get a reference to the job
js = jobs(25).subjobs[8]

# Make a copy of the subjob as a real job
j = js.copy(unprepare=True)

# Give the job a convenient name so you remember what you are doing
j.name = '25.8'

# Change backend to Interactive [or Local()]. This works even if the data is not at CERN.
j.backend=Interactive()
# Add a statement to the options increasing debug level
j.application.extraopts = 'YourAlgo.OutputLevel=3;'
# note that <YourAlgo> will be <DaVinciInit()> if DaVinci is your application

# If more extraopts necessary the syntax is:
j.application.extraopts = '''YourAlgo.OutputLevel=3;

# Run it interactively
j.submit()

**Can I submit Ganga jobs from the LSF batch system?**

Yes, but why would you want to?

You can let LSF take care of the job submission by using the following shell script (ganga.csh):

```bash
#!/bin/tcsh
setenv X509_USER_PROXY $HOME/private/grid.proxy
SetupProject Ganga
ganga ganga.py
```

where ganga.py is a python script that creates, configures and submits the ganga job.

Before submitting ganga.csh to the batch system you have to place your grid proxy in your AFS home area (which is visible from any LSF worker node) instead of on the local machine's /tmp area by executing the following two commands (see also How do I run jobs that require a grid proxy on a LSF):

```bash
setenv X509_USER_PROXY $HOME/private/grid.proxy
lhcb-proxy-init
```

**How to avoid the CPU time limit on lxplus machines?**

- Install Ganga locally
- Really, install Ganga locally
- Use a virtual machine like CERNVM® to run Ganga locally
- Make sure you’re using the latest version of ganga which is usually the fastest

**Correct platform setting, or, "What do I do if some of my DIRAC jobs fail with Could not retrieve Tool?" (28 June 2019)**

If some of your jobs are mysteriously unable to retrieve a tool, it could be that you have set the wrong platform for your jobs. Not all nodes have CentOS 7 available; test your job interactively on an lxplus6 machine (ssh lxplus6.cern.ch): If you get the same error as your failed DIRAC job, this is the likely culprit.

The default platform is `x86_64-slc6-gcc62-opt`, which tells Ganga it’s okay to run on nodes with SLC6 installed but not necessarily CentOS 7. (Note also that a given site [job.backend.actualCE=] can have some nodes with CentOS 7 and some without.) This can be a problem _even if you called `make` x86_64-slc6-gcc62-opt_. There are three solutions:

1. `make` your application on lxplus6, making sure you first called `LbLogin -c 86_64-slc6-gcc62-opt`. Your application should be able to run on all worker nodes. Submit your jobs again. (Not `resubmit`; see, "How do I make a new job that runs only on the inputdata from some failed subjobs?")
2. `make` your application on lxplus7 using a CentOS 7 platform, e.g., `x86_64-centos7-gcc62-opt`, and set the platform for your application before submitting, e.g., `MyApplication.platform =`
Ganga Set up/Configuration

How do I define and load utility functions from within ganga? (16 Nov 2009)

The simplest way of doing this is to write a python module containing your utility functions and classes. Save this file as ~/.ganga.py and it will be automatically sourced.

If you need to write a script or module in a more stand alone way, the ganga GPI objects can be imported from Ganga.GPI, as long as the ganga libraries are on your PYTHONPATH.

How to use Ganga functionality from a Python script

You can import Ganga as a module. See the following example

First in bash (this could also be done using os.environ inside the Python script; has to be done before the first Ganga import though).

    export GANGA_CONFIG_PATH=GangaLHCb/LHCb.ini
    export GANGA_SITE_CONFIG_AREA=/cvmfs/lhcb.cern.ch/lib/GangaConfig/config
    export PYTHONPATH=$PYTHONPATH:/cvmfs/ganga.cern.ch/Ganga/install/LATEST

    /lib/python2.7/site-packages/

and then in Python

    import ganga.ganga

    # Everything is now in the "ganga" namespace
    j = ganga.Job(backend=ganga.Dirac())
    j.submit()

    ganga.runMonitoring()

How do I use Root within Ganga?

Make sure you use the same version of root for your merging/processing/browsing as you used to create the files!

Here is an example for DaVinci v25r7p1

Root is setup/configured in three places:

1. Your bashrc/tcshrc
2. When you do SetupProject something
3. In your .gangarc (config.ROOT) (for the Root backend)

So there are several things you need to do.

Correct platform setting, or, "What do I do if some of my DIRAC jobs fail with Could not retrieve Tobj?" (28 June 2019)
Choose the version of root which corresponds to the application you are using.

```
> SetupProject DaVinci
> which root #only an example
/afs/cern.ch/sw/lcg/app/releases/ROOT/5.26.00cp1_python2.5/slc4_ia32_gcc34/root/bin/root
```

edit your .barchrc/tcshrc to use that version:

```
> export ROOTSYS=/afs/cern.ch/sw/lcg/app/releases/ROOT/5.26.00cp1_python2.5/slc4_ia32_gcc34/root
```

edit your .gangarc to use that version:

```ini
[defaults_Root]
version = 5.26.00cp1_python2.5 #only an example

[ROOT]
version = 5.26.00cp1_python2.5 #only an example
```

SetupProject for Ganga with the right version of Root

```
> SetupProject Ganga ROOT -v 5.26.00cp1_python2.5 #only an example
```

Root merging fails, segfaults recursively or takes an infinite amount of time or memory or writes a file of infinite size to disk

Root is not known for backwards compatibility, it is likely you have created the ntuple with a different version of root that with which you try to merge it. See above.

**How do I change where my gangadir is?**

You can set this to be anywhere you have write access by setting the [Configuration] gangadir field in your ~/.gangarc file.

**How can I run the monitoring loop for a subset of jobs?**

If you start Ganga without the monitoring turned on (ganga --no-mon), you can monitor a job slice as you like using the runMonitoring method.

```
slice = jobs.select(name='FooBar')
runMonitoring(jobs=slice)
```

should run the monitoring once for all jobs called 'FooBar'.

**How do I set the Ganga output location for a cluster outside of CERN?**

You should most likely use the SharedFile object instead of the MassStorageFile object. You will need to configure this first in the defaults_SharedFile section of your .gangarc file or provide it in a common INI file for your cluster.

**Data Sets and Data Files**

**How do I make a dataset that only contains the files added since my last run**

How do I use Root within Ganga?
Let's say you have the data you have already analysed in an LHCbDataset `ds1`. All the data (including what you have already analysed) is in `ds2`. Then to create a dataset `ds3` with just the new files you can do:

```python
ds3 = LHCbDataset (files=[f for f in ds2.files if f not in ds1.files])
```

or using the simple command:

```python
ds3 = ds2.difference(ds1)
```

**How do I make a dataset only containing data taken between specific runs?**

If the data you want to analyse was taken in a specific run period you may configure your job to run only over this data.

This may be used in combination with other conditions, such as magnet polarity up or down.

### March 2011 data sets ###

```python
bk_march = BKQuery(
    dqflag = "OK",
    path = "87223-87977/Real Data/Reco09/Stripping13/90000000/CHARMCOMPLETEEVENT.DST",
    type = "Run"
)
```

```python
ds_march = bk_march.getDataset()
```

### Magnet down data sets ###

```python
bk_full_magdown = BKQuery(
    dqflag = "OK",
    path = "/LHCb/Collision11/Beam3500GeV-VeloClosed-MagDown/RealData/Reco09/Stripping13/90000000/CHARMCOMPLETEEVENT.DST",
    type = "Path"
)
```

```python
ds_full_magdown = bk_full_magdown.getDataset()
```

### get only magnet down runs taken in March ###

```python
ds_march_magdown = ds_march.intersection(ds_full_magdown)
```

```python
j.inputdata = ds_march_magdown
j.submit()
```

**How do I use the outputdata from one Grid job as the inputdata for another?**

First, get the LHCbDataset of LFN's for the completed grid (i.e. Dirac) job:

```python
d = j.backend.getOutputDataLFNs()
```

Then, simply set this as the inputdata for the new job:

```python
new_job.inputdata = d.
```

If you want to collect all of the outputdata from a collection of subjobs, then do:

```python
ds = LHCbDataset()
for sj in j.subjobs.select(status="completed"):
    ds.extend(sj.backend.getOutputDataLFNs())
new_job.inputdata = ds
```

**NOTE** that at present the above doesn't work (see https://github.com/ganga-devs/ganga/issues/930). The code below should certainly work though:

```python
ds = LHCbDataset()
for sj in j.subjobs.select(status=?completed?):
    lfn = []
    for lfn in sj.backend.getOutputDataLFNs():
```

How do I make a dataset that only contains the files added since my last run?
lfns.append('LFN:' + lfn.lfn)
ds.extend(lfns)
new_job.inputdata = ds

How do I make a new job that runs only on the inputdata from some failed subjobs?

Collect the inputdata from the failed subjobs:

ds = LHCbDataset()
for sj in j.subjobs.select(status="failed"):
    ds.extend(sj.inputdata)

Then copy the original job and reset its inputdata to the files you've just collected:

j = j.copy()
j.inputdata = ds

How do I extract an LHCbDataset from a Gaudi-style options file?

Simply choose the Gaudi application for which the options file was written (e.g. DaVinci), and do: ds = DaVinci().readInputData('/path/to/options.py').

Why am I getting a "No ancestors found" error?

Most likely you have the depth attribute of your LHCbDataset set incorrectly. It should be 0 if the file was created by you (or another user) and not via production. For Ganga 5.4.0 and later, depth=0 is the default but for earlier releases depth=1 was the default.

How do I deal with replicas (e.g. replicate, get replicas, remove replicas) in Ganga?

For lfn = DiracFile(lfn='/some/lfn.file'), you can do:

lfn.removeReplica('CERN-USER') # remove replica at, e.g., CERN-USER
lfn.replicate('CERN-USER') # replicate file at, e.g., CERN-USER

To obtain a list of SE's, do lfn.replicate().<verbatim>

To obtain a list of SE's, do lfn.replicate().

How do I download a file from Grid storage?

Simply do df.get() to download to your home directory or df.get('/path/to/download/to') where df = DiracFile(lfn='/some/lfn.file'). See help(DiracFile.get) for more info. Note: To get all the DiracFiles from a DIRAC job, you can do:

for sj in j.subjobs:
    for f in sj.outputfiles.get(DiracFile):
        f.localDir = sj.outputdir
        f.get()
How do I remove a file from Grid storage?

Simply do `lfn.remove()` where `lfn = DiracFile('/some/lfn.file')`. **WARNING!** This removes ALL replicas of the file!

How to I remove all files from a Dirac job?

The produce is the same as above, but you need to get an LHCbDataset containing the output LFNs of the job:

```python
for sj in j.subjobs:
    for f in sj.outputfiles.get(DiracFile):
        if f.lfn:
            f.remove()
```

**WARNING!** This removes ALL replicas of the file!

How do I upload a file to Grid storage?

Simply do `pfn.put(uploadSE=['CERN-USER'])` where `pfn = DiracFile('/some/pfn.file')`. This returns a DiracFile object which can be used to replicate, etc. the new LFN.

```python
d = DiracFile('parrot.txt')
d.put()
```

or for wildcard files

```python
file_list = DiracFile('*.txt').put()
```

How can I put a file bigger than 10 MB in my input sandbox for a Dirac job?

First you need to upload the file to grid storage. So, e.g., you can do `pf = DiracFile('/some/path/my.file', lfn='/the/lfn/my.file')` and upload it by doing `pf.put(uploadSE='CERN-USER')`. You can also upload any file using the Dirac command line tools.

Once the file is uploaded, simply add it to the `job.inputfiles` list for your job. E.g. do `j.inputfiles = [DiracFile(lfn='/the/lfn/my.file')]`. Then submit the job as normal. When the job runs, Dirac will download the file into the work dir and you can access it as simply `my.file`.

How do I run on SDSTs?

**This is in principle obsolete as we do not produce SDSTs any more.** SDSTs do not contain all the information for DaVinci to run. You need to access the raw file. In the general case this is only possible at CERN, so make sure the dataset you want to analyse is at CERN (or that the SDST and RAW files are readable for users). Select your dataset from the bookkeeping interface (for instance Greig’s test sample is run 77222: use the run lookup button). Then make sure you set `j.inputdata.depth=2`. This will add the required file catalog allowing to read in the ancestor files.
Streaming DST from different SE (CERN)

It is possible to read (stream) a dst file at CERN in your Ganga job using the interactive backend. This can be useful for developers who need to read up to a few hundred events. This does not work at present if the DST you are looking for does not exist at CERN. The Ganga configuration is as follows:

a. Set the "site" to CERN in the local Ganga configuration:

```python
config.LHCb.LocalSite="LCG.CERN.ch"
```

b. For the job, then you need the following options for example:

```python
backend=Local()
inputdata = LHCbDataset (files = [DiracFile(lfn='/lhcb/MC/MC10/ALLSTREAMS.DST/00009117/0000/00009117_00000686_1.allstreams.dst')])
analysis.setupProjectOptions = '--use-grid'
```

The job should now be set up to read the dst from CERN. The advantage of this is to avoid copying data locally for short bits of development.

**How to run over locally-stored DST?**

You can use local DST by uploading them as LocalFiles:

```python
j.inputdata = [LocalFile('/path/to/file.dst'),LocalFile('/path/to/other/file.dst')]
```

**How to select LHCb dataset using Bookkeeping information (for example TCK)?**

```python
def FilterTCKs(path="/LHCb/Collision11/Beam3500GeV-VeloClosed-MagUp/Real Data/Reco12/Stripping17/90000000/EW.DST",badtcks=['0x740036']):
  dataset = BKQuery(path).getDataset()
  datasetM = BKQuery(path).getDatasetMetadata()
  newLFNs = []
  for f in dataset:
    if not datasetM['Value'][f.lfn]['TCK'] in badtcks: #this is the actual filter criterion
      newLFNs.append(f.lfn)
  returnDs = LHCbCompressedDataset(newLFNs)
  return returnDs
```

There were two main considerations that went into this snippet that need to be taken into account when deriving a different selection from this example:

- A query for the metadata takes O(1 s) time so it should be done for the dataset and not for each file.

**What are the pros and cons of MassStorageFile and DiracFile?**

There are two methods of setting the output of a Ganga job on the Dirac backend to be stored on the EOS disk storage system. You can set the output file using MassStorageFile, via:

```python
j.outputfiles = [MassStorageFile('NtupleName.root')]
```

This stores the output in a "private" EOS area. Alternatively, you can set the output file using DiracFile, via:

```python
j.outputfiles = [DiracFile(namePattern='NtupleName.root',locations=["CERN-USER"])]
```
This stores the output in a CERN storage element. These two methods have some pros and cons that you should consider.

**MassStorageFile**

The `MassStorageFile` specification tells Ganga that the output should be stored in the user's private EOS area at CERN.

- **Pro:** You can use the disk space provided by EOS.
- **Pro:** You can open up files directly in ROOT from here while running at CERN.
- **Con:** The client where the job runs can't write directly to your private EOS area. For this reason, the output passes through the Ganga client. This is inefficient and error prone.

**DiracFile**

The `DiracFile` specification stores the output in a CERN storage element. This can be confusing, as EOS is the underlying storage solution here as well.

- **Pro:** Output is uploaded directly from the client.
- **Pro:** More stable, as fallback solutions are applied if it fails the first time.
- **Pro:** Can be read directly from ROOT.
- **Con:** Counts towards your "Dirac" quota instead.

For stability reasons, the second solution (using `DiracFile`) is recommended. There are, of course, situations where the `MassStorageFile` solution has advantages (mostly when not at CERN).

**Retrieving/merging output**

**How do I turn off/on default downloading of the output sandbox for failed jobs?**

The default behavior is "on" (i.e. to download the sandbox). This behavior is controlled by the `config.LHCb.failed_sandbox_download` option. Set it to False for "don't download" and True for "download" (set it in your .gangarc file for permanent changes).

To set it in your gangarc modify or add the line under:

```
[LHCb]
failed_sandbox_download=False
```

**How do I download an output sandbox by hand?**

Do `j.backend.getOutputSandbox()` and the sandbox will appear in the job's `outputdir`.

**How do I control output file locations? (3 Dec 2008)**

This gives a brief discussion on how to control where output files go, for a more details see the user's guide.

The output of a job can end up:

- In castor (or other grid SE) [see "outputdata" below]
- In output sandbox (gangadir/etc etc) [see "outputsandbox" below]
The "outputsandbox" is meant to only contain small files (typically 1 MB or less). Larger files should go to "outputdata". The standard output and standard error are automatically placed in the output sandbox. The user can completely control the destination of all output files for any individual job using the j.outputfiles and j.outputdata fields. For Gaudi jobs, files created by the various services are automatically collected and sorted into the sandbox and data. The user can control how the default sorting is done by altering the outputsandbox_types field in their .gangarc file. The default behavior can also be overridden in any individual job using the j.outputfiles and j.outputdata fields (for example for having the root file in the outputdata do j.outputdata = ['*.root']). For more details see section 4.5 of the manual.

Note: When using the DIRAC backend, all files over 10 MB will go to SE regardless of what the user requests in Ganga.

As of Ganga 6, the distinction between outputsandbox and outputdata has been removed and as such these attributes are now deprecated and cannot be used (remain solely for legacy reasons). There is now a single output attribute called outputfiles. It is the type of object that is attached to this attribute that determines it's final location. LocalFile objects end up back with the user as if in the old outputsandbox attribute, while DiracFile objects will be uploaded to SE.

**How do I retrieve outputdata (large files)? (27 Jan 09)**

Job output files redirected to SE (castor, cern) are uploaded to wherever DataOutput (in your .gangarc file) points. This defaults to castor for jobs submitted at CERN. The output can be retrieved using the getOutputData method on the Dirac backend. From ganga 5.1.5 there will also be an additional method on the backend to give the explicit LFNs of outputdata. You should refer to the online help in ganga for more details - help(Dirac).

From Ganga 6, jobs that specify a DiracFile as part of their outputfiles attribute will (when they complete) have a fully populated DiracFile object which contains Lfn and guid information. To download the file one merely needs to use the get() method of the DiracFile.

For convenience one can now filter the outputfiles list based on either file namepattern or file type so downloading a large number of files can be as simple as:

- e.g. matching all files on namePattern
  
  ```python
  for f in job.outputfiles.get("*.root"):  
    f.get()
  ```

- or matching a particular instance of DiracFile
  
  ```python
  for f in job.outputfiles.get(DiracFile("A.root")):  
    f.get()
  ```

- or matching all files by type
  
  ```python
  for f in job.outputfiles.get(DiracFile):  
    f.get()
  ```

**Do the mergers only work for local files? (4 Jun 08)**

Yes. No plans to implement anything else at the moment until we are convinced it was really useful. We would still have to download the parts to a local disk, merge them and then upload the results.
Can I prevent stdout from being in my output sandbox/Can I put stdout in output data?

No. Dirac does not allow this; thus, Ganga does not allow it either. Adding "stdout" to outputdata or outputfiles cause problems (so don't do it). You can delete it once it's downloaded if you want. Grid jobs should not be run in verbose mode to save space.

Bookkeeping

How do I get an LHCbDataset from the bookkeeping GUI in Ganga?

Simply do `data = browseBK()` and select the files you want. When you save them the GUI will close and the data set will be saved in the `data` object.

How do I get a "path" for the BKQuery object?

From the GUI, you can do the following: start the GUI from inside Ganga; dig down to the dataset you want; right click and select bookmark; you can then cut and paste the path from the window opening up. Once you have the path you can get updated data sets from the BKQuery object w/o having to use the GUI.

How can I find out the tags / DDDB tags / project versions / options for a given BK selection?

Data in the BK always corresponds to a particular processing pass which describes how the data was produced. This is most readily accessible in the BK GUI (at the time of writing this is not yet available on the BK web page) by "right-clicking" on the processing pass name and selecting "more information":

This provides all the information in the BK about how the selected data was produced e.g.
Since there are many LHCb mac users I include the below X11 preferences that should be checked in order to be able to "right-click" in the BK GUI:

```
Emulate three button mouse
- Hold Option or Command while clicking to activate the middle or right mouse buttons.

Follow system keyboard layout
- Allows input menu changes to overwrite the current X11 keymap.

Enable key equivalents under X11
- When enabled, menu bar key equivalents may interfere with X11 applications that use the Meta modifier.
```

This topic: LHCb/FAQ > GangaLHCbFAQ
Topic revision: r110 - 2020-07-18 - MarkElliotSmith

Copyright &© 2008-2020 by the contributing authors. All material on this collaboration platform is the property of the contributing authors. Ideas, requests, problems regarding TWiki? Send feedback