Table of Contents

Overview

An LHCb software project is basically a collection of packages that, optionally, depend on other projects.

A project is described by a CMT description file (`project.cmt`), a CMake description file (`CMakeLists.txt`), a technical file for CMake (`toolchain.cmake`) and a container package (usually called `<Project>Sys`).

The CMT description file (`project.cmt`)

The `project.cmt` file is located in the directory `cmt` under the top level directory of the project. It contains the name of the project, the projects it depends on and some configuration options. This file will become obsolete once we are fully migrated to CMake

An example of `project.cmt` is:

```
1 project LHCB
2
3 use GAUDI GAUDI_v23r8
4 use DBASE
5 use PARAM
6
7 build_strategy with_installarea
8 setup_strategy root
```

The name of the project must appear in uppercase letters (line 1). The dependencies on other projects should include the version of the project (if any) in the form `<PROJECT>_vXrY` (line 3).

The CMake description file (`CMakeLists.txt`)

The `CMakeLists.txt` file is located in the top level directory of the project. It is equivalent to `cmt/project.cmt` and should be kept synchronised with it, it describes the project structure and dependencies for use by the CMake build tools. It contains the name of the project (line 9, capitalised lower case), the projects it depends on (line 10), and any data packages used by packages contained in the project (lines 11-14).

An example of `CMakeLists.txt` is:

```
 1 CMAKE_MINIMUM_REQUIRED(VERSION 2.8.5)
 2
 3 #---------------------------------------------------------------
 4 # Load macros and functions for Gaudi-based projects
 5 find_package(GaudiProject)
 6 #---------------------------------------------------------------
 7
 8 # Declare project name and version
 9 gaudi_project(LHCb v36r0
10               USE Gaudi v23r8
11               DATA Gen/DecFiles
12                    Det/SQLDDDB VERSION v7r*
13                    FieldMap
14                    TCK/HltTCK
15                    TCK/L0TCK VERSION v4r*)
16
```

The `toolchain.cmake` file

This is a technical file needed to bootstrap CMake. It is the same for all projects, so can be copied from another project when creating the project, and does not need to be touched again

The Container Package

The container package defines the contents of the project. It contains instructions for building the project with CMT and CMake, release notes and instructions for building the documentation with doxygen, and instructions for running tests. The directory structure, including all required files is shown below (taking the NOETHER project as an example)

```
CMakeLists.txt

cmt:
requirements

doc:
DoxyFile.cfg  DoxyNoether.cfg  MainPage.h  release.notes

tests/options:
configurables.py

tests/qmtest/noethersys.qms:
configurables.qmt

tests/refs:
configurables.ref
```

It is recommended that for a new project you copy the above structure and files from an existing project and edit as necessary.

Note that the name of a package must be unique among all the packages in all the projects.

Repository Structure and Tags

As described in GaudiSVNRepository, the Subversion repository contains one top-level directory per project with the same name of the project (correct case). That directory features the three standard directories `trunk`, `tags` and `branches`.

The `trunk` directory contains a complete image of the project main development version. It contains the `cmt` directory, with the latest revision of the file `project.cmt`, and all the packages belonging to the project, starting from the container package.

The packages have their own tags in the `tags` directory under a sub-directory with the same name of the package (e.g. `$GAUDISVN/Gaudi/tags/GaudiKernel/v27r7`).

The project tags have the format `<PROJECT>_vXrY` and are under `tags/<PROJECT>`. The project tags is basically a tag of the project `cmt` directory (e.g. `$GAUDISVN/Gaudi/tags/GAUDI/GAUDI_v21r7/cmt`), but it can include the actual versions of all the packages (copies from their tags as in, e. g., the tag of Gaudi).

Note that the container package and the project must be tagged with the same version (e.g. `tags/LHCbSys/v36r0` and `tags/LHCB/LHCB_v36r0`). If the project contains an application, also the application package should normally be tagged with the same version as the project.

Adding a New Project

The necessity for creation of a new project should first be discussed and agreed at the Physics Applications Coordination (PAC) meeting. A Project Manager should be identified who will be expected to maintain the project and in particular coordinate requests for releases through the PAC meeting.

For the Project Manager

Video walkthru to accompany this text (filmed during the creation of the MooreOnline project, if it doesn't load, hit refresh):

- Create the project structure in the repository (`tcsh`)
    - ♦ prepare the structure in a local directory

```
cd $TMPDIR
set proj = Newproject
set PROJ = `echo $proj | tr a-z A-Z`
svn co -N svn+ssh://svn.cern.ch/reps/lhcb
cd lhcb
svn mkdir $proj $proj/trunk $proj/tags $proj/branches $proj/trunk/cmt $proj/tags/$PROJ
```

- ♦ create the file `$proj/trunk/cmt/project.cmt` with a content like

```
project $PROJ

# Add one or more lines for the project dependencies, dependency on LHCB_v36r0 given as an exampl
use LHCB LHCB_v36r0

build_strategy with_installarea
setup_strategy root
```

- ♦ create the file `$proj/trunk/CMakeLists.txt` with a content like

```
CMAKE_MINIMUM_REQUIRED(VERSION 2.8.5)

#---------------------------------------------------------------
# Load macros and functions for Gaudi-based projects
find_package(GaudiProject)
#---------------------------------------------------------------

# Declare project name and version and dependencies (LHCB v36r0 given as example)
gaudi_project($proj v1r0
              USE LHCb v36r0)
```

- ♦ create the file `$proj/trunk/toolchain.cmake` by copying it from another project

```
cp $LHCBRELEASES/LHCB/LHCB_v36r0/toolchain.cmake $proj/trunk/toolchain.cmake
```

- ♦ put the files `project.cmt`, `CMakeLists.txt` and `toolchain.cmake` under the control of Subversion, commit and remove the temporary directory

```
svn add $proj/trunk/cmt/project.cmt
svn add $proj/trunk/CMakeLists.txt
svn add $proj/trunk/toolchain.cmake
cd ..
```

```
svn ci -m "Project structure for project $proj" lhcb
rm -rf lhcb
```

- Create the container package `<Project>Sys` following the instructions at CreateNewPackageSVN.
    - It is mandatory for the package to contain the files `CMakeLists.txt` at the top level, `requirements` in `cmt` and `release.notes`, `DoxyFile.cfg`, `MainPage.h` in `doc`
    - It is highly recommended to provide also a `tests` directory structure and files as in the NOETHER example above
- Add both the project and the package to the `projects` and `packages` property of the repository:

```
cd $TMPDIR
svn co -N svn+ssh://svn.cern.ch/reps/lhcb
svn propedit projects lhcb
svn ci -m "Added the project NewProject to the list of projects" lhcb
rm -rf lhcb
```

The instructions for the `packages` property are described in CreateNewPackageSVN.

For the Librarian

- The creation of a new project needs a full new release of the LbScripts project. It cannot be dynamically added (yet):
- The project name has to be added to `$LBCONFIGURATIONROOT/python/LbConfiguration/Project.py`. This will ensure the proper support for the various aliases for the project. If the Project name length is > 6 then adjust the AFS volume main name in that same file. This will be the name used for the volume of each release of the new project.
- create an afs volume called `<PROJECT>` in `$LHCBTAR`. Do not forget to add the correct ACLs, including `system:anyuser read`
- create directory called `<PROJECT>` in `$LHCBRELEASES`. Do not forget to add the correct ACLs, including `system:anyuser read`
- Add the project to the DOC directories: see http://cern.ch/LHCb-release-area/DOC/documentation.html and ask the AFS administrator to add the project manager to the AFS group (You need to create the directory /afs/cern.ch/lhcb/software/releases/DOC/ for mkproject to succeed)
- Add the project to the `Field Label: Project/Package` in the `Tasks/Edit Fields Values` menu of https://savannah.cern.ch/task/?group=lhcbdeployment

Deprecated: This step used to be necessary but the role of this scripts is now performed by Project.py:

- For backward compatibility with install_project.py, the file `$LBLEGACYROOT/python/LbLegacy/LHCb_config.py` has to be edited. The project name has to be added to its internal list.

This topic: LHCb > AddProject
Topic revision: r15 - 2013-09-11 - RobLambert