Table of Contents

How to Use it

- how to use it

Overview

- install_project is used to install a Project and all its dependencies in a directory retreived by:
  - setenv MYSITEROOT `pwd`
- the script can install only the sources or the sources and binaries:
  - *python install_project.py -p <Project> -v <version>*
- the binary is, by default, the $CMTCONFIG one unless another one is specified:
  - *python install_project.py -p <Project> -v <version> -b*
- different binaries can be installed on the same tree.
  - *python install_project.py -p <Project> -v <version> --binary=<another_binary>*
  - it is possible to install a win32 binary from a Linux box, but, in this case, it is not possible to build vsnet files.
- data files (XlmDDDB, SQLDDDB, Geant4Files, ParamFiles, DecFiles) can be installed in the same way:
  - *python install_project.py -p XmlDDDB -v <version>*
- the current version of CMT is installed by default, another one can be installed on request:
  - *python install_project.py -p <Project> -v <version> -c <another_cmt_version>*
- the script is available from the web. It has a version number of the type *yymmdd*
  - the latest version is always download and the version compared to the local one. If the local one is older than the latest the script is stopped.
- The script download a script tar file which contains the most commonly used LHCb scripts.
- The script download a system tar file with some libraries requested for the Production on the GRID, one of them is *libshift.so* .
- A html file is associated to every LHCb Project or data file tar files.
- When a tar file has be *untarred* with success it is replaced with a dummy tar file to save disk space.
- If a tar file cannot be *untarred* it is remove with its associated html file.
- The html file contains the list of dependent Project tar files.
- The script tries to NOT overwrite anything already installed:
  - if a tar file already exists it is not download.
  - if a project_version already exists the corresponding tar file is not *untarred*.
- To re-installed a Project it is **necessary** to remove it first (html, tar.gz, Project_version tree):
  - *python install_project.py -p <Project> -v <version> -r*
  - *python install_project.py -p <Project> -v <version> -b*
- To re-installed some external package first remove the LCGCMT version it belongs to and the external package:
  - *python install_project.py -p <LCGCMT> -v <lcg_version> -r*
  - *python install_project.py -p <external_package> -v <ext_version> -r*
  - *python install_project.py -p Gaudi -v <version_using_the_LCGCMT> -b*
- To get the list of available versions of a Project:
  - *python install_project.py -p<Project> -l*

Usage

```
print ' install_project.py - version %s:  install a project in the current directory '%(script_
print """
Usage:
  cd <somewhere>
  setenv MYSITEROOT $PWD
  python install_project.py -p <project> -v <version> [-b| --binary=<bin> ] [-d or --debug] [-m

      $CMTCONFIG  is the binary directory name
  creates log/         to receive log files
          scripts/     to receive script files
          lcg/         to receive lcg software
          lhcb/        to receive lhcb software
          contrib/     to receive CMT and OpenScientist
          targz/       to receive tar files from the web
          $CMTCONFIG/  to receive runtime libraries
  download necessary scripts in scripts/
  get the list of projects to download
  download project sources
  if binaries are required: download project binaries
  otherwise compile project sources
  -d or --debug      : to print more info
  -l or --list       : to list the <project>_<version>_*.tar.gz files available on the web
  -r or --remove     : remove the <project>/<version>
  -c or --cmtversion : download this CMT version
  -m do_config       : to make a 'cmt broadcast cmt config' of all projects but LCGCMT
  -m global          : to compile all projects but LCGCMT
  -m select          : to compile only the project given in -p argument
  -h or --help       : to print this help
  -b                 : to import $CMTCONFIG binaries
  --binary=<bin>     : to import another binary (i.e. $CMTCONFIG_dbg on Linux)
                       this triggers a 'cmt broadcast cmt config' of all projects but LCGCMT
  -f                 : to import source, $CMTCONFIG binaries, $CMTCONFIG_dbg binaries and
                       to make a 'cmt broadcast cmt config' of all projects but LCGCMT

Perequisite:
  requires python version >= 2.3.4 on Win32 and python >=2.2.3 on Linux
  the following environment variables should have been set before invoking the script:
        $MYSITEROOT is the full path of the current directory
        $MYSITEROOT is not a link.
        > cd <somewhere>
        > setenv  MYSITEROOT $PWD
        $CMTCONFIG  is the binary directory name
        if you want to download binaries $CMTCONFIG should be identical to
        one the LHCb CERN platforms
Remarks:
  scripts and system tar.gz files are always download
  project tar.gz files are not download if they already exist.
  tar.gz file which cannot be untared is removed and a message is printed
  """
```

Full Description

functions are in alphabetic order but *usage* and *help* and *main*

- **main** reads the command line, decodes the arguments, set flags, gets LHCb_config.py, calls run_install.
- **run_install** executes the installation:
  - ♦ ◊ check definition of $MYSITEROOT and $CMTCONFIG
    - · exit if not defined
  - ◊ **create _dir** the structure below $MYSITEROOT, return logfile descriptor
    - · html_dir, targz_dir, log_dir, contrib_dir, lhcb_dir, lcg_dir, script_dir, system_dir
  - ◊ if require **list_versions** gives the list of available versions and exit
  - ◊ if require **remove_project** removes a project_version and exit
  - ◊ **get_scripts** tar file, overwrite existing one and untar it
  - ◊ **get_CMT** tar file and install it if not already there, return the CMT version
  - ◊ **set_lhcb_env** ironment
  - ◊ download source tar files first
    - · **get_project_list**
      - from project name and version build the file name (html or tar.gz)
        - ♦ <PROJECT><PROJECT>_<version>
      - get the html file from the web if it is not already there and read it:
        - ♦ to fill a html_list which contains the list of tar files to download
        - ♦ to build a project_list dictionary which for each tar file gives the origin:
          - ◊ {'<PROJECT><PROJECT>_<version>.tar.gz' : '<PROJECT>' , 'LCGCMT_<vers>_<binary>.tar.gz' : 'source' , etc ...}
      - return the html_list and the project_list dictionary
    - · **get_project_tar**
      - tar_list keys are file names, tar_list values are the location directories (source/, system/, <PROJECT>/)
        - ♦ tar files from source/ directory are pure binary tar files: the right binary file must be chosen according to CMTCONFIG value. these tar files are *untarred* either in lcg_dir (LCGCMT, GENSER) or in contrib_dir.
        - ♦ tar files from <PROJECT>/ directory are *untarred* in lhcb_dir.
        - ♦ tar file from system/ directory is *untarred* in $CMTCONFIG.
          - ◊ **get_file**
            - · decide if it is necessary to download the file (.tar.gz, .html, .py)
              - scripts and system tar files are always overwritten if they already exist.
              - other projects are not overwritten.
                - ♦ lhcb projects can exist as source but not binary , or a binary exists but not the required one:
                  - ◊ check existence of InstallArea/<binary> in the project path

- if it exists return with exist_flag = True
- external projects do not have an InstallArea, but an empty <binary> directory.
  - check existence of <binary> directory in the project path.
    - if it exists return with exist_flag = True
- if exist_flag = False
  - download the file with *urllib.urlretrieve*, retry once, exit if it does not work.
- untar_flag = **get_untar_flag** which returns *yes* if exist_flag = False
- if untar_flag == 'yes'
  - **untar_file**
    - if python_version > 2.3.4 but not 2.4.3 (which seems to not work properly)
      - use the *tarfile* python module.
    - if *tarfile* module not available : exit on win32, extract with *tar* on Linux
    - to save disk space remove the tar file, replace it with a one_line text file.
    - in case of error, remove the faulty tar file and exit.
  - if it is an external project (one from ext_lhcb list with no source, only binary) creates an empty binary directory in lcg_dir/project/vers to remember which binary tar file has been installed.
  - if it is an lhcb project binary file make sure that InstallArea/binary directory exists, if not create it
  - on Linux update the softlinks (which point to the afs release area) with the LbInstallArea shell script
- **cmt_configure** the project and all its dependencies, returns *application* package paths if any.
  - loop over all projects in the html_list order, the first one is the required project
    - from the file name **get_pack_ver** returns in a list: PROJECT name, PROJECT_version, binary, fullpath * if binary installation has been requested:
  - **get_project_list** of binaries
  - **get_project_tar** binary tar files

- if the project is an *application* project then run **exec_cmt_library_links** to update the library links in the application package.

· if one of the following options (do_config, select, global) has been requested call **compile_project**

- **compile_project**
  - ♦ if CMTPROJECTPATH is set then use it to set CMTPATH otherwise CMTPATH has been set earlier.
  - ♦ loop over all paths of CMTPATH but LCGCMT, DBASE and PARAM
    - ◊ to execute *cmt br cmt config*
    - ◊ if *do_config*
      - · rebuild library links on all packages
      - · on linux if InstallArea exists reset softlinks.
    - ◊ if *select*
      - · compile only the just installed project
    - ◊ if *global*
      - · compile all projects one at a time.

· if the full option has been requested:

- move $CMTCONFIG to binary_dbg
- **get_project_list** of binary_dbg project
- **get_project_tar** of binary_dbg tar files
- **compile_project** with do_config option

-- FlorenceRanjard - 30 Mar 2007

This topic: LHCb > InstallProject
Topic revision: r12 - 2007-04-25 - FlorenceRanjard