Bayeux/trunk installation report on macOS 10.9.5

F.Mauger, C.Augier

2017-02-21

This note reports how we have built and installed Bayeux/trunk on a macOS 10.9.5 system for the EDELWEISS-DM experiment. Links:

- https://github.com/SuperNEMO-DBD/brew.git

Notes:

- CadfaelBrew is only supported on 64-bits systems.
- Two build systems are supported : GNU/make and Ninja, on top of which CMake is used to build Bayeux.

Contents

The target system

- Architecture:

  ```
  $ uname -a
  Darwin ... x86_64
  ```

- Environment:

  The system must have a relatively *bare* environment. It means that even if a lot of software has been installed on the system (`/usr/bin`) or in some alternative locations (`/usr/local`, `/opt`...), you should be able to run a shell with a lightweight `PATH`, typically something like:

  ```
  $ echo $PATH
  /opt/X11/bin:/usr/bin:/bin:/usr/sbin:/sbin:/sw/bin:/usr/local/mysql/bin:/Users/{LOGIN}/loca
  ```

  In principle, you should not have the `LD_LIBRARY_PATH` environmental variable set:

  ```
  $ echo "aaa${DYLD_LIBRARY_PATH}ZZZ"
  aaaZZZ
  ```

Important notice: You have to check carefully both environment variables above because it is frequent that some system administrators use them to setup by default some third party software. The keyword here is *by default* which means something you didn't ask for. Unfortunately, the excessive/improper usage of these environs (mostly `DYLD_LIBRARY_PATH`) may ends to conflict while building CadfaelBrew and/or Bayeux.

- Dependencies:

  It is be mandatory (or maybe useful) to install additional system packages to properly build Bayeux and activate some of its features. This is documented below.

Setup of CadfaelBrew

Links:

- CadfaelBrew repository (GitHub, public access)
- Cadfael (SuperNEMO Wiki, private access)

Please follow the instructions on the installation report at https://nemo.lpc-caen.in2p3.fr/browser/Bayeux/trunk/doc/InstallationReports/Cadfaelbrew/macOS10.9.5/tagged/cadfaelbrew_macos10.9.5_report-1.0.pdf

brew

Once you have installed CadfaelBrew, you should be able to run a *brew* session:

```
$ brewsh      # Enter a brew shell
...
$ echo $PATH
.../CadfaelBrew/bin:...

$ which brew
.../CadfaelBrew/bin/brew
```

This opens a new shell with all environmental variables activated to setup all the software tools managed through CadfaelBrew (utilities, compiler(s), Boost, Root, Geant4...).

You can check the location and version of core software utilities:

```
$ which cmake
.../CadfaelBrew/bin/cmake
$ cmake --version
cmake version 3.7.2

$ which clang
.../CadfaelBrew/.../clang
$ clang --version
Apple LLVM version 6.0 ...
...

$ brew install doxygen
$ which doxygen
.../CadfaelBrew/bin/doxygen
$ doxygen --version
1.8.12
```

Ninja

Ninja is a build system which can be used in place of (GNU)make. Install Ninja through `brew` if it was not already done before (you must setup the brew environment for that):

```
$ brew install ninja
...
```

Then you can check your Ninja version:

```
$ which ninja
.../CadfaelBrew/bin/ninja
$ ninja --version
1.7.2
```

Qt5

Not supported yet.

Configuration and build of Bayeux/trunk

Links:

- Bayeux (SuperNEMO Wiki, private access)

System dependencies

Install dependencies and useful utilities:

```
$ brew install docutils
```

Working directory

Set the software base directory where there is enough storage capacity to host Bayeux (> 1 GB).
You may adapt this base directory to your own system, for example:

```
$ mkdir ${HOME}/softs/EDELWEISS/Bayeux
$ cd ${HOME}/softs/EDELWEISS/Bayeux
```

Then create a few working directories:

```
$ mkdir Source  # hosts the source directories
$ mkdir Binary  # hosts the build/installation directories
```

Download Bayeux

Download Bayeux/trunk source files:

```
$ cd Source
$ export BX_SOURCE_BASE_DIR="$(pwd)"
$ svn co https://nemo.lpc-caen.in2p3.fr/svn/Bayeux/trunk Bayeux-trunk
$ cd Bayeux-trunk
$ export BX_DEV_SOURCE_DIR="$(pwd)"
$ LANG=C svn info
...
Revision: 18593
...
```

Configure Bayeux

1. Create a build directory and cd in it:

```
$ cd ../../Binary
$ mkdir Bayeux-trunk
$ cd Bayeux-trunk
$ export BX_DEV_BIN_DIR="$(pwd)"
$ export BX_DEV_BUILD_DIR="${BX_DEV_BIN_DIR}/Build-macOS"
$ mkdir -p ${BX_DEV_BUILD_DIR}
$ cd ${BX_DEV_BUILD_DIR}
$ pwd
.../Bayeux/Binary/Bayeux-trunk/Build-macOS
```

2. Configure the Bayeux build with CMake and using Ninja and GCC :

```
$ export BX_DEV_INSTALL_DIR="${BX_DEV_BIN_DIR}/Install-macOS"
$ cmake \
 -DCMAKE_BUILD_TYPE:STRING="Release" \
 -DCMAKE_INSTALL_PREFIX:FILEPATH="${BX_DEV_INSTALL_DIR}" \
 -DBAYEUX_CXX_STANDARD="11" \
 -DBAYEUX_COMPILER_ERROR_ON_WARNING=OFF \
 -DBAYEUX_WITH_IWYU_CHECK=ON \
 -DBAYEUX_WITH_DEVELOPER_TOOLS=ON \
 -DBAYEUX_WITH_LAHAGUE=ON \
 -DBAYEUX_WITH_GEANT4_MODULE=ON \
 -DBAYEUX_WITH_MCNP_MODULE=OFF \
 -DBAYEUX_WITH_QT_GUI=OFF \
 -DBAYEUX_ENABLE_TESTING=ON \
 -DBAYEUX_WITH_DOCS=ON \
 -DBAYEUX_WITH_DOCS_OCD=ON \
 -GNinja \
 ${BX_DEV_SOURCE_DIR}
```

Build

Using 8 processors to go faster (depends on your machine):

```
$ ninja -j8
...
```

Test programs

Before to do the final installation, we run the test programs:

```
$ ninja test
[1/1] Running tests...
Test project /opt/sw/Bayeux/Binary/Bayeux-trunk/Build-gcc-cxx11-ninja-Linux-x86_64
        Start   1: datatools-test_reflection_0
  1/326 Test   #1: datatools-test_reflection_0 .......   Passed    0.10 sec
...
        Start 343: bxbayeux-test_bayeux
343/343 Test #343: bxbayeux-test_bayeux ..............   Passed    0.07 sec

100% tests passed, 0 tests failed out of 326

Total Test time (real) =  68.23 sec
```

Installation

Run:

```
$ ninja install
...
```

Check installation

Browse the installation directory:

```
$ LANG=C tree -L 3 -F ${BX_DEV_INSTALL_DIR}
...
```

Suggestions for a Bash setup (see below)

Here we explicitely *load/setup* the Bayeux environment from a Bash shell with a dedicated function defined in my `~/.bashrc` startup file:

```
echo >&2 "[info] Type 'bayeux_dev_setup' to use Bayeux/trunk..."
export BX_BASE_DIR="${HOME}/softs/EDELWEISS/Bayeux"
export BX_DEV_BIN_DIR="${BX_BASE_DIR}/Binary/Bayeux-trunk"
export BX_DEV_BUILD_DIR="${BX_DEV_BIN_DIR}/Build-macOS"

# The Bayeux/trunk setup function:
function do_bayeux_trunk_setup()
{
 if [ -z "${CADFAELBREW_INSTALL_DIR}" ]; then
   echo >&2 "[error] Not a CadfaelBrew shell ! Please run 'brewsh'!"
   return 1
 fi
 if [ -n "${BX_DEV_INSTALL_DIR}" ]; then
   echo >&2 "[error] Bayeux/trunk is already setup !"
   return 1
 fi
 export BX_DEV_INSTALL_DIR="${BX_DEV_BIN_DIR}/Install-macOS"
 export PATH="${BX_DEV_INSTALL_DIR}/bin:${PATH}"
 echo >&2 "[notice] Bayeux/trunk is now setup !"
 return 0;
}
export -f do_bayeux_trunk_setup
alias bayeux_dev_setup="do_bayeux_trunk_setup"
```

When one wants to use pieces of software from Bayeux, one runs:

```
$ brewsh
{brew/shell}$ bayeux_dev_setup
{brew/shell}$ which bxquery
...
```

Update the source code from the Bayeux/trunk

1. Activate the CadfaelBrew environment:

   ```
   $ brewsh
   ```

2. Cd in the Bayeux/trunk source directory:

   ```
   {brew/shell}$ cd ${HOME}/softs/EDELWEISS/Bayeux/Source/Bayeux-trunk
   ```

3. Update the source code:

   ```
   {brew/shell}$ svn up
   ```

4. Cd in the Bayeux/trunk build directory:

   ```
   {brew/shell}$ cd ${BX_DEV_BUILD_DIR}
   ```

5. You may need to clean the build directory:

   ```
   {brew/shell}$ ninja clean
   ```

   and even to completely delete and rebuild it from scratch:

   ```
   {brew/shell}$ cd ${BX_DEV_BUILD_DIR}
   {brew/shell}$ rm -fr *
   ```

5. You may need to delete the install tree:

   ```
   {brew/shell}$ rm -fr ${BX_DEV_BIN_DIR}/Install-gcc-cxx11-Linux-x86_64
   ```

6. Reconfigure (see above):

   ```
   {brew/shell}$ cmake ...
   ...
   ```

6. You may need to delete the install tree:

   ```
   {brew/shell}$ rm -fr ${BX_DEV_BIN_DIR}/Install-macOS
   ```

7. Rebuild, test and (re)install:

   ```
   {brew/shell}$ ninja -j8
   {brew/shell}$ ninja test
   {brew/shell}$ ninja install
   ```