Установка ROOT (Data Analysis Framework)

Подробное описание (с комментариями) процесса установки программного пакета ROOT на систему Ubuntu 16.04LTS.

Официальное краткое руководство по установке: https://root.cern.ch/building-root

Требования к окружению

Зависит от операционной системы, подробней тут: https://root.cern.ch/build-prerequisites

Для Ubuntu требуется:

```
sudo apt-get install git dpkg-dev cmake g++ gcc binutils \
libx11-dev libxpm-dev libxft-dev libxext-dev python-dev
```

python-dev в скрипте выше не является необходимым компонентом, но рекоммендую установить его для возможности использовать функционал root из python.

Есть ещё опциональные пакеты необходимые для определенного функционала ROOT (подробности - по ссылке выше).

Скачиваем исходники

Подробная инструкция по скачиванию: https://root.cern.ch/get-root-sources

Последняя версия доступна в репозитории. Для клонирования ветки master из консоли создаем и заходим в папку, куда надо сохранить исходники и запускаем команду:

```
git clone http://github.com/root-project/root.git
```

Для установки определенной версии необходимо в скачанном репозитории откатить изменения до желаемой версии. Релизы в репозитории ROOT'а отмечены специальными тегами, так что искать нужную версию в списке всех коммитов не придется:

```
# Получить список тегов:
git tag -l

# Создать ветку от определенного тега (версии)
git checkout -b <new_brunch_name> <version>
```

где `<new_brunch_name>` - название новой ветки проекта (рекоммендуется называть по номеру версии).

`<version>` - название тега (версии).

Мы создаем новую ветку в репозитории, чтобы работать с ней (вносить изменения в файлы) и при этом иметь возможность вернуться к последней версии root (и обратно) простым переходом в ветку master без необходимости скачивать репозиторий заново.
Мы создаем новую ветку в репозитории, чтобы работать с ней (вносить изменения в файлы) и при этом иметь возможность вернуться к последней версии ROOT (и обратно) простым переходом в ветку master без необходимости скачивать репозиторий заново.

Два подхода к установке

- Через **cmake**: подходит для любой операционной системы, современный подход, будет поддерживаться в дальнейшем.
- Через связку **configure, make, install**: подходит для linux, простой и быстрый способ, может не поддерживать некоторый новейший функционал ROOT'а (но это не точно :).

Установка по фиксированному пути

При этом варианте ROOT установится в систему по указанному пути или по умолчанию (для linux: программа в `/usr/local/`, библиотеки в `/usr/local/lib`) и будет доступен для всех пользователей без необходимости создания дополнительных переменных среды при каждом перезапуске системы или консоли.
При этом варианте root установится в систему по указанному пути или по умолчанию (для linux: программа в `/usr/local/`, библиотеки в `/usr/local/lib`) и будет доступен для всех пользователей без необходимости создания дополнительных переменных среды при каждом перезапуске системы или консоли.

Установка проходит в три стадии: генерация скриптов, компилирование (build), установка компонентов.

Генерация скриптов

Для linux доступен классический вариант - через `./configure` с указанием необходимых параметров. Подробней: https://root.cern.ch/build-root-old-method
Для linux доступен классический вариант - через `./configure` с указанием необходимых параметров. Подробней: https://root.cern.ch/build-root-old-method

Рекомендуется кроссплатформенный вариант - через `cmake`.
Сначала создаем пустую папку, куда будут записаны установочные скрипты и впоследствии скомпилированы исходники.
Потом переходим в эту папку (в консоли) и оттуда запускаем одну из следующих команду:

Вариант 1. Подготовка скриптов для установки по выбранному пользователем пути:

```
cmake <path/to/root/source> -DCMAKE_INSTALL_PREFIX=<install_path> -Dgnuinstall=ON
```

- `<install_path>` - это папка, в которую необходимо установить ROOT
- `<path/to/root/source>` - путь до исходников (репозитория)
- параметр `-Dgnuinstall=ON` включает режим установки по фиксированному пути без него файлы будут скомпилированы и готовы к работе, но не будут установлены в систему

Вариант 2. Подготовка скриптов для установки в папку по умолчанию:

```
cmake <path/to/root/source> -Dgnuinstall=ON
```

Компилирование и установка

Самый **ДОЛГИЙ** процесс. Занимает **МНОГО** часов.

Из папки с установочными скриптами запускаем команду от имени администратора (суперпользователя):

```
sudo cmake --build . --target install -- -j<N>
```

где `<N>` - это количество потоков обработки (оптимально 3 для двуядерного процессора с 2 потоками на ядро).

Затем обновлям кэш ldconfig:

```
ldconfig
```

Пробуем запустить ROOT:

```
# стандартный запуск
root

# запуск без экрана приветствия/информации
root -l
```

Возможные ошибки

Если ROOT не запускается, стоит проверить правильность прописанных путей до библиотек:

```
ldd /usr/local/root/bin/rootcint
```

Путь до программы `rootcint` может отличаться в зависимости от указанного при установке ROOT пути.

Должно выводить пути до библиотек, например:

```
libCint.so.5.06 => /usr/local/root/lib/libCint.so.5.06 (0x00002aaaaaaad000)
libdl.so.2 => /lib64/libdl.so.2 (0x0000003f68200000)
libstdc++.so.6 => /usr/lib64/libstdc++.so.6 (0x0000003f6a400000)
libm.so.6 => /lib64/libm.so.6 (0x0000003f68000000)
libgcc_s.so.1 => /lib64/libgcc_s.so.1 (0x0000003f6a200000)
libc.so.6 => /lib64/libc.so.6 (0x0000003f67d00000)
/lib64/ld-linux-x86-64.so.2 (0x0000003f67b00000)
```

Если пишет `<lib_name> => Not found`, значит путь до библиотеки `<lib_name>` не прописан. Придется использовать переменные среды PATH и LD_LIBRARY_PATH, которые задаются скриптом:

```
source <path_to_build_dir>/bin/thisroot.sh
```

где `<path_to_build_dir>` - папка, в которую скомпилировали исходники.

Если без выполнения этого скрипта ROOT не запускается, то его (скрипт) придется запускать при каждой перезагрузке терминала.

Шпаков Константин, август 2018