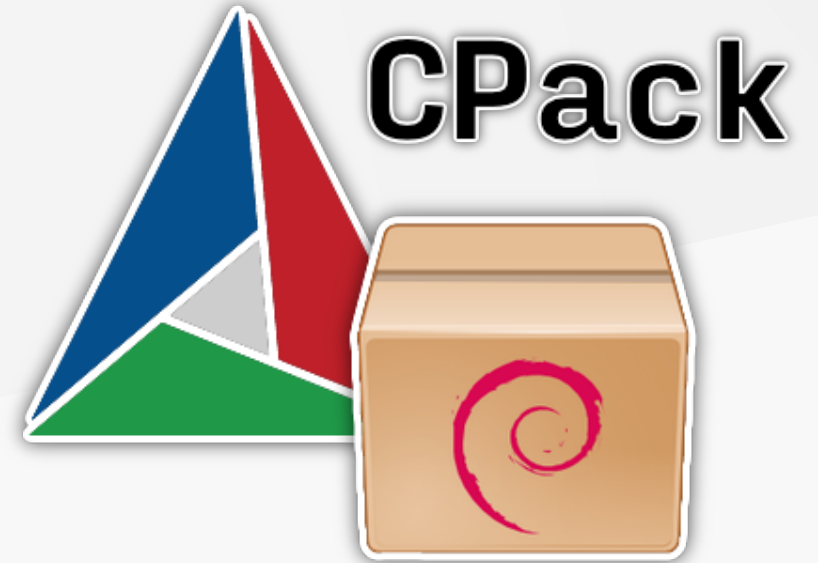




Packaging with CMake

Creating packages for Debian, RedHat & FreeBSD with CPack (CMake)





TOC

- Introduction
- Goals
- Usage
 - Source Compile
 - Static File Packaging
- Output
- Hints



Introduction

- CMake is a Cross Platform Make (platform independent)
- Has been created as a part of the Visible Human Project
- Creates projects by Scriptfiles (CMakeLists.txt) and Makefiles
- Support various development tools:
 - Borland Makefiles
 - MinGW/MSYS/Nmake/Unix Makefiles
 - Visual Studio
 - XCode
- CMake includes DART, CDash, CTest, CPack



Goals

Focussing on packaging we want to

- Use CPack
- Create ready to use packages for
 - Debian
 - RedHat
- Create packages independent of your platform (e.g. on macOS for Debian)
 - May vary

```
#[stable(feature = "rust1", since = "1.0.0")]
impl<'a, T, P> Iterator for Split<'a, T, P>
where
    P: FnMut(&T) -> bool,
{
    type Item = &'a [T];

    #[inline]
    fn next(&mut self) -> Option<&'a [T]> {
        if self.finished {
            return None;
        }

        match self.v.iter().position(|x| (self.pred)(x)) {
            None => self.finish(),
            Some(idx) => {
                let ret = Some(&self.v[..idx]);
                self.v = &self.v[idx + 1..];
                ret
            }
        }
    }

    #[inline]
    fn size_hint(&self) -> (usize, Option<usize>) {
        if self.finished { (0, Some(0)) } else { (1, Some(self.v.len() + 1)) }
    }
}
```



Usage: Why?

- Easy to use
- Single file
- All options for both distributions in a single place
- Easy CI/CD integration



Usage: Source Compile

- Packaging a compiled source file only needs one additional file
- Packages automatically as .deb and .rpm is needed

```
.  
├─ CMakeLists.txt  
└─ helloworld.c  
  
0 directories, 2 files
```



Usage: Source Compile

```
cmake_minimum_required(VERSION 3.16)
project>HelloWorld VERSION 0.1)

add_executable(hello-world main.cc)
install(TARGETS hello-world RUNTIME DESTINATION bin)
set(EXE_PATH ${CMAKE_INSTALL_PREFIX}/bin/hello-world)

set(CPACK_PACKAGE_VERSION ${CMAKE_PROJECT_VERSION})
set(CPACK_GENERATOR "RPM")
set(CPACK_PACKAGE_NAME "hello-world")
set(CPACK_PACKAGE_RELEASE 1)
set(CPACK_PACKAGING_INSTALL_PREFIX ${CMAKE_INSTALL_PREFIX})
include(CPack)
```



Usage: Static File Shipping

- Packaging a binary file only needs one additional file
- Skipping the source compile
- Packages automatically as .deb and .rpm is needed

```
.  
├─ CMakeLists.txt  
└─ helloworld.py
```

```
0 directories, 2 files
```




Usage: Static File Shipping

```
cmake_minimum_required(VERSION 3.16)
project(ChangelogFragmentsCreator VERSION 1.1)

install(FILES helloworld.py DESTINATION bin)

set(CPACK_PACKAGE_VERSION ${CMAKE_PROJECT_VERSION})
set(CPACK_GENERATOR "RPM")
set(CPACK_PACKAGE_NAME "hello-world")
set(CPACK_PACKAGE_RELEASE 1)
set(CPACK_PACKAGING_INSTALL_PREFIX ${CMAKE_INSTALL_PREFIX})
include(CPack)
```



Usage: Changes

What has changed?

```
add_executable(hello-world main.cc)
```



Options for Redhat & Debian

```
cmake_minimum_required(VERSION 3.16)
project(ChangelogFragmentsCreator VERSION 1.1)

install(FILES ../changelog-creator DESTINATION bin)

# General
set(CPACK_RESOURCE_FILE_LICENSE "${CMAKE_CURRENT_SOURCE_DIR}/../LICENSE")
set(CPACK_RESOURCE_FILE_README "${CMAKE_CURRENT_SOURCE_DIR}/../README.md")
set(CPACK_DEBIAN_PACKAGE_MAINTAINER "Florian Paul Azim Hoberg <gyptazy@gyptazy.ch>")
set(CPACK_PACKAGE_CONTACT "Florian Paul Azim Hoberg <gyptazy@gyptazy.ch>")
set(CPACK_PACKAGE_VENDOR "gyptazy")

# RPM packaging
set(CPACK_PACKAGE_VERSION ${CMAKE_PROJECT_VERSION})
set(CPACK_RPM_PACKAGE_ARCHITECTURE "noarch")
set(CPACK_PACKAGE_NAME "hello-world")
set(CPACK_RPM_PACKAGE_SUMMARY "hello-world - A hello world output.")
set(CPACK_RPM_CHANGELOG_FILE "${CMAKE_CURRENT_SOURCE_DIR}/rpm_changelog.txt")
set(CPACK_RPM_PACKAGE_LICENSE "GPL 3.0")
set(CPACK_RPM_PACKAGE_REQUIRES "python >= 3.2.0, python3-pyyaml")

# DEB packaging
set(CPACK_DEBIAN_FILE_NAME DEB-DEFAULT)
set(CPACK_DEBIAN_PACKAGE_ARCHITECTURE "noarch")
set(CPACK_DEBIAN_PACKAGE_SUMMARY "hello-world - A hello world output.")
set(CPACK_DEBIAN_PACKAGE_CONTROL_EXTRA "${CMAKE_CURRENT_SOURCE_DIR}/deb_changelog.txt")
set(CPACK_DEBIAN_PACKAGE_REQUIRES "python3-yaml")
set(CPACK_DEBIAN_PACKAGE_LICENSE "GPL 3.0")

# Install
set(CPACK_PACKAGING_INSTALL_PREFIX ${CMAKE_INSTALL_PREFIX})
include(CPack)
```



Usage: Creating packages

Creating packages is easy! Therefore, we just need to run one command for each package type.

```
cpack -G RPM .  
cpack -G DEB .
```



Output

```
├─ hello-world-1.0-noarch.rpm
├─ hello-world_1.0_noarch.deb
├─ _CPack_Packages
│   └─ Linux
│       └─ DEB
│           ├── hello-world-1.0-noarch
│           │   ├── control
│           │   ├── control.tar.gz
│           │   ├── data.tar.gz
│           │   ├── deb_changelog.txt
│           │   ├── debian-binary
│           │   ├── md5sums
│           │   └─ usr
│           │       └─ local
│           │           └─ bin
│           │               └─ changelog-creator
│           └─ hello-world_1.0_noarch.deb
│   └─ RPM
│       ├── BUILD
│       ├── BUILDROOT
│       ├── hello-world-1.0-Linux
│       │   ├── usr
│       │   │   ├── local
│       │   │   │   └─ bin
│       │   │       └─ changelog-creator
│       ├── rpmbuildhello-world.err
│       ├── rpmbuildhello-world.out
│       ├── RPMS
│       │   └─ hello-world-1.0-noarch.rpm
│       ├── SOURCES
│       ├── SPECS
│       │   ├── hello-world.spec
│       │   └─ hello-world.spec.in
│       ├── SRPMS
│       └─ tmp
```



Hints

Still take care of some things

- Distribution specific things
 - Changelog format
 - Package names for dependencies
 - Package name of your created package
- CPack as a dependency



Resources

Examples

- Changelog Fragments Creator:
<https://github.com/gyptazy/changelog-fragments-creator/>
- Compile & shipping pre/post scripts:
<https://github.com/andrew-hardin/cpack-systemd-demo/>

Presentation

- This presentation: https://gyptazy.ch/talks/cmake_cpack/



Thanks!

Web: <https://gyptazy.ch>

Twitter: @gyptazy

Fediverse: gyptazy@bsd.cafe

