



INTRODUCTION TO CONAN 2.0

MEETING C++ TOOL FAIR

Christopher McArthur, Conan Developer Advocate



Introduction!



Christopher McArthur

Former Conan User and long time contributor

“I’ve been opening enterprise issues for Conan over the [past 3 years](#)”



@prince_chrismc



prince-chrismc



Chris Mc#6297



Chris Mc U017JHD34KB

Conan Team Socials



@conan_io



conan-io



CONAN

C/C++ Package Manager

What is Conan?



CONAN

C/C++ Package Manager



C and C++ Package Manager

What is the role of a package manager?

- Easily install dependencies
 - `conan install`
`--requires=spdlog/1.11.0`

How is Conan different?

- Enable you to build and distribute binaries

We offer **JFrog's ConanCenter** with 1500+ open-source projects with over 100 configuration (os, compiler, arch) being created and publish to help accelerate open-source.

- Open-Source (MIT license)
- Distributed (1 client many servers)
- Scalable and flexible
- Remotes + Server

Supports

- CMake, Meson, Autotool, etc...
- Any Platforms
- More than just CMakeLists.txt



CONAN

C/C++ Package Manager

Conanfile: A simple consumer

app/1.0

```
$ git clone ... app && cd app  
$ conan install .
```

conanfile.txt

```
[requires]  
spdlog/1.11.0  
  
[generators]  
CMakeToolchain  
CMakeDeps  
  
[layout]  
cmake_layout
```

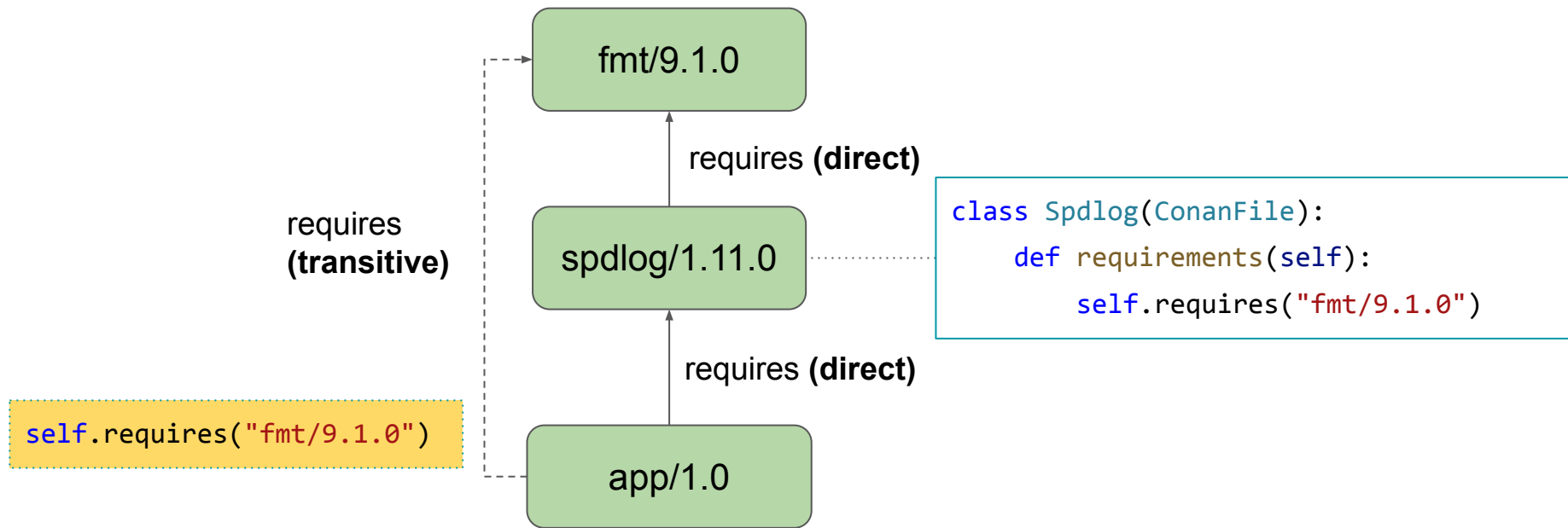


CONAN

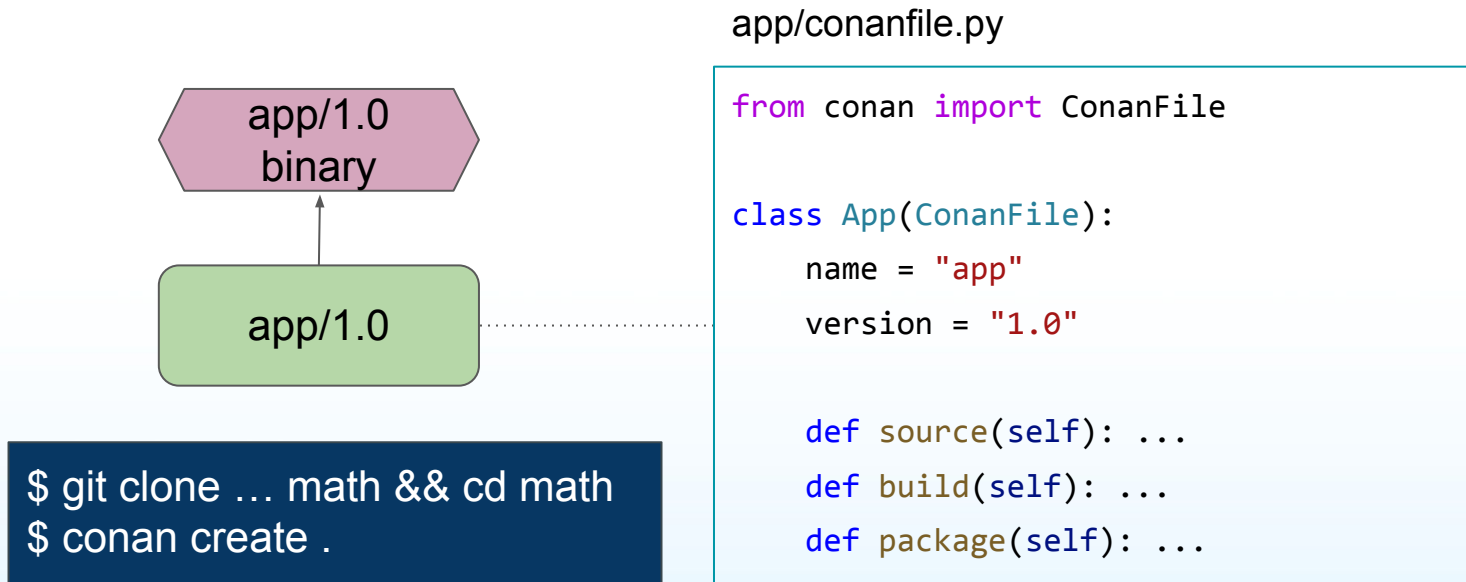
C/C++ Package Manager



Dependencies - Graph Model



Conanfile: A package “recipe”



Demo





CONAN

C/C++ Package Manager

Introduction to the new Tutorial Section

<https://docs.conan.io/2/tutorial.html>

**CONAN 2.0**
C/C++ Package Manager

Search docs

Introduction

What's new in Conan 2.0

Install

☰ **Tutorial**

Consuming packages

Creating packages

Working with Conan repositories

Developing packages locally

Versioning

Other important Conan features

Integrations

Examples

Reference

Knowledge

Changelog

Docs » Tutorial

Edit on GitHub

Tutorial

The purpose of this section is to guide you through the most important Conan features with practical examples. From using libraries already packaged by Conan, to how to package your libraries and store them in a remote server alongside all the precompiled binaries.

- **Consuming packages**
 - Build a simple CMake project using Conan
 - Using build tools as Conan packages
 - Building for multiple configurations: Release, Debug, Static and Shared
 - Understanding the flexibility of using conanfile.py vs conanfile.txt
 - How to cross-compile your applications using Conan: host and build contexts
 - Introduction to versioning
- **Creating packages**
 - Create your first Conan package
 - Handle sources in packages
 - Add dependencies to packages
 - Preparing the build
 - Configure settings and options in recipes
 - Build packages: the build() method
 - Package files: the package() method
 - Define information for consumers: the package_info() method



CONAN

C/C++ Package Manager

Why Conan?

- Key difference between Conan and other C/C++ package manager is the focus on binaries. Being able to create packages that can be **re-used** across multiple teams throughout an organization. Enables flexibility and scalability. Framework for doing DevOps and Package Management in a very enterprise ready manner.

You can model **platform configurations** and the **linkage between libraries**. To deterministically know what to build but more importantly what you already have to save time and money!



CONAN

C/C++ Package Manager

What's new in Conan 2.0



CONAN

C/C++ Package Manager

Everything is new!



1.0

5 years, without breaking

60% new code, 20%
backports

1.X \Leftrightarrow 2.0 compatible syntax
subset



2.0

What to expect?

- Limited installer support - more to come
- Default home is now `~/.conan2`
- Cache layout changes
 - Temporary build folder
- Changes to `settings.yml`
- Opt-in default profile
- Always using two profiles, statically provided
- Unified command reference syntax
 - Lots of command line changes
- Generators
- Environments and scopes
- Structured outputs, serializations, custom formatters
- Private and hidden requirements by default
- Handling conflicts resolution
- **New graph resolution**
- Requirement traits
- Package types
- Binary Model and package IDs
- **Extensions**
 - Hooks
 - Profiles Checker
 - Command Wrapper
 - Package Signing
 - Compatibility
 - Deployers
 - Custom Commands
- Python API
- **Lockfiles**
- User settings



CONAN

C/C++ Package Manager

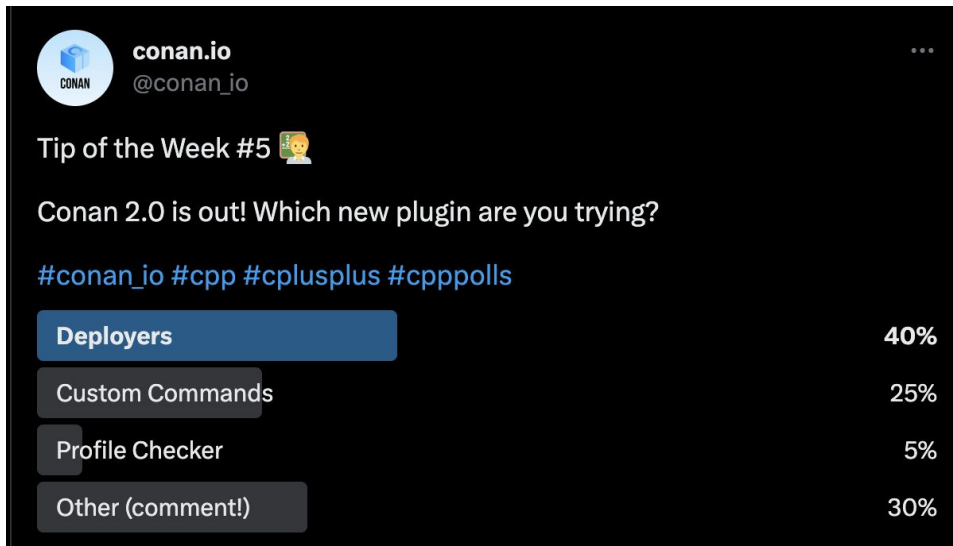
Everything is on docs.conan.io

Plugins

The solution - empower users to do it themselves!

Provide a framework for users to build solutions tailored to their needs with mechanisms that give them controlled management.

- Profile Checker
- Command Wrapper
- Package Signing



CONAN

C/C++ Package Manager

What's new?



New graph

New plugin extensions

New deployers

New binary compatibility

Multi-revision cache

package_id

Lockfiles

New configuration and environment

Package immutability optimizations

... and many more

<https://docs.conan.io/2/whatsnew.html>



CONAN

C/C++ Package Manager

More Resources?

- ACCU Talk by Diego about the graph improvements:
<https://youtu.be/kKGglzm5ous>
- Introducing Conan 2.0 blog post:
<https://blog.conan.io/2023/02/22/Conan-2.0.html>
- New tutorial section <https://docs.conan.io/2/tutorial.html>
- Questions?
 - <https://github.com/conan-io/conan/issues>
 - <https://cppalliance.org/slack/> (community help)

Keep up to date! https://twitter.com/conan_io



CONAN

C/C++ Package Manager

Conclusion



`pip install conan`

<https://conan.io>



CONAN

C/C++ Package Manager