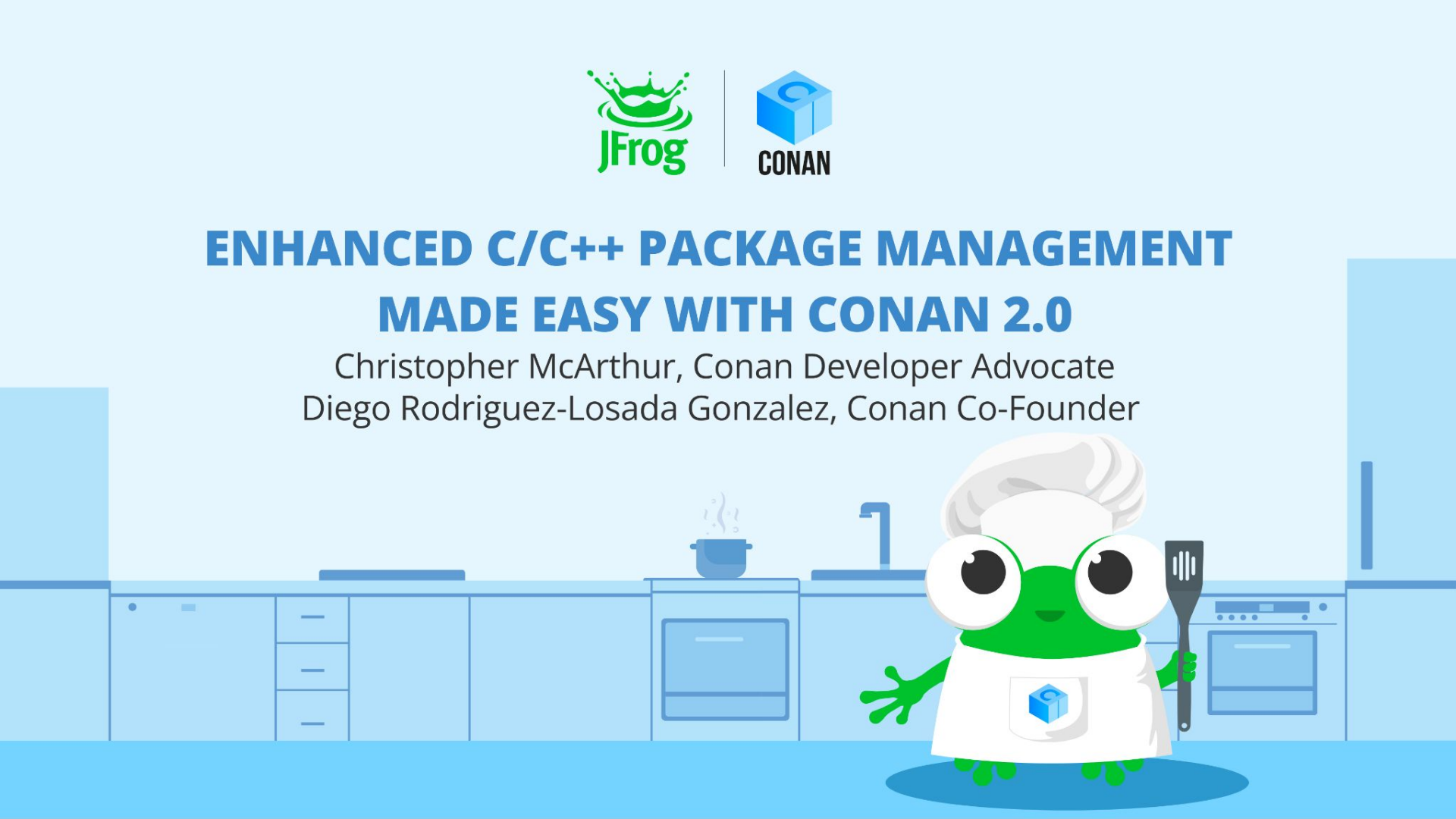


ENHANCED C/C++ PACKAGE MANAGEMENT MADE EASY WITH CONAN 2.0

Christopher McArthur, Conan Developer Advocate
Diego Rodriguez-Losada Gonzalez, Conan Co-Founder

Introduction!

Christopher McArthur

Former Conan User and long time contributor

“I’ve been opening enterprise issues for Conan over the past 3 years”

Diego Rodriguez-Losada Gonzalez

Conan Co-Founder and Maintainer

“I have been resolving enterprise issues for Conan over the past 6 years”

JFrog

CONAN

C/C++ Package Manager

What is Conan?

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.

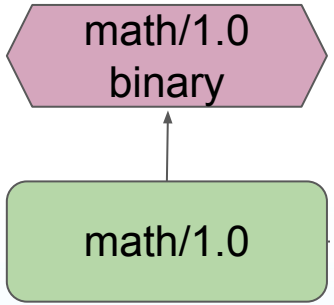
- Open-Source (MIT license)
- Distributed (1 client many servers)
- Scalable and flexible
- Remotes + Server

Supports

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

JFrog | CONAN C/C++ Package Manager

Conanfile: A package “recipe”




```
$ git clone … math && cd math
$ conan create .
```

math/conanfile.py

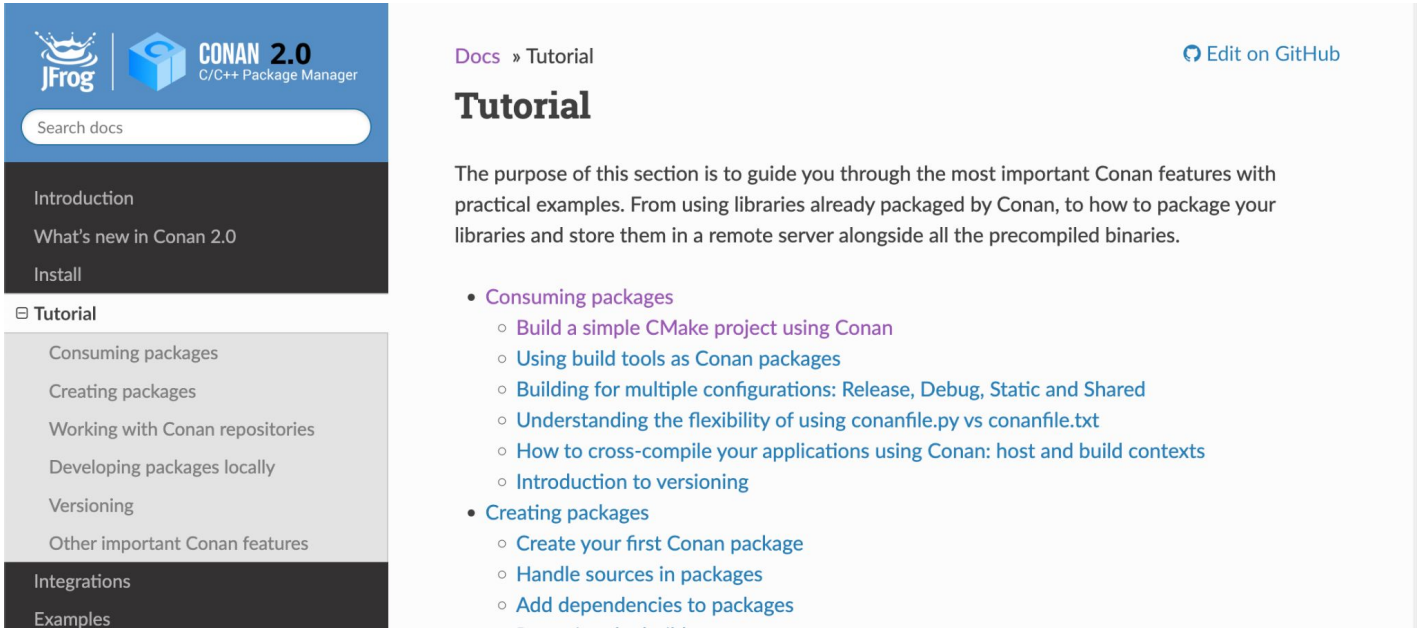
```python
from conan import ConanFile

class Math(ConanFile):
    name = "math"
    version = "1.0"

    def source(self): ...
    def build(self): ...
    def package(self): ...
```


Introduction to the new Tutorial Section

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



Why Conan?

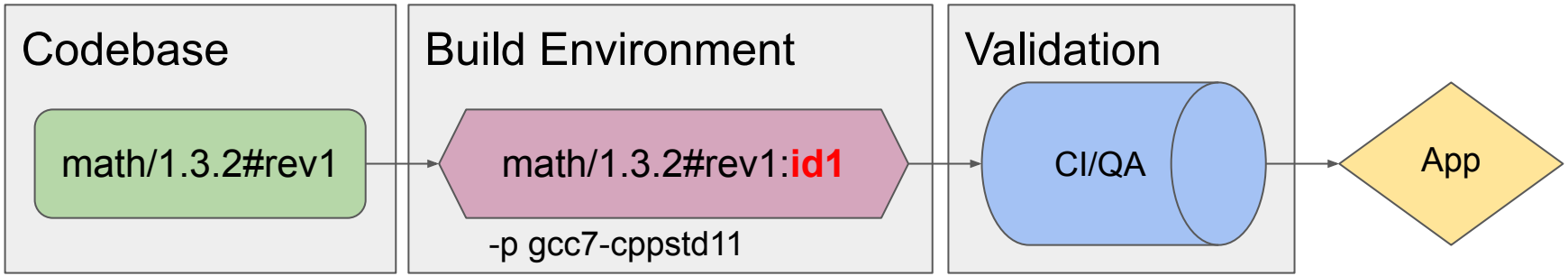
- Key difference between Conan and other C/C++ package manager is the focus on binaries. Being able to create packages that can be **re-used** across multiple teams throughout an organization. Enables flexibility and scalability. Framework for doing DevOps and Package Management in a very enterprise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

A common problem

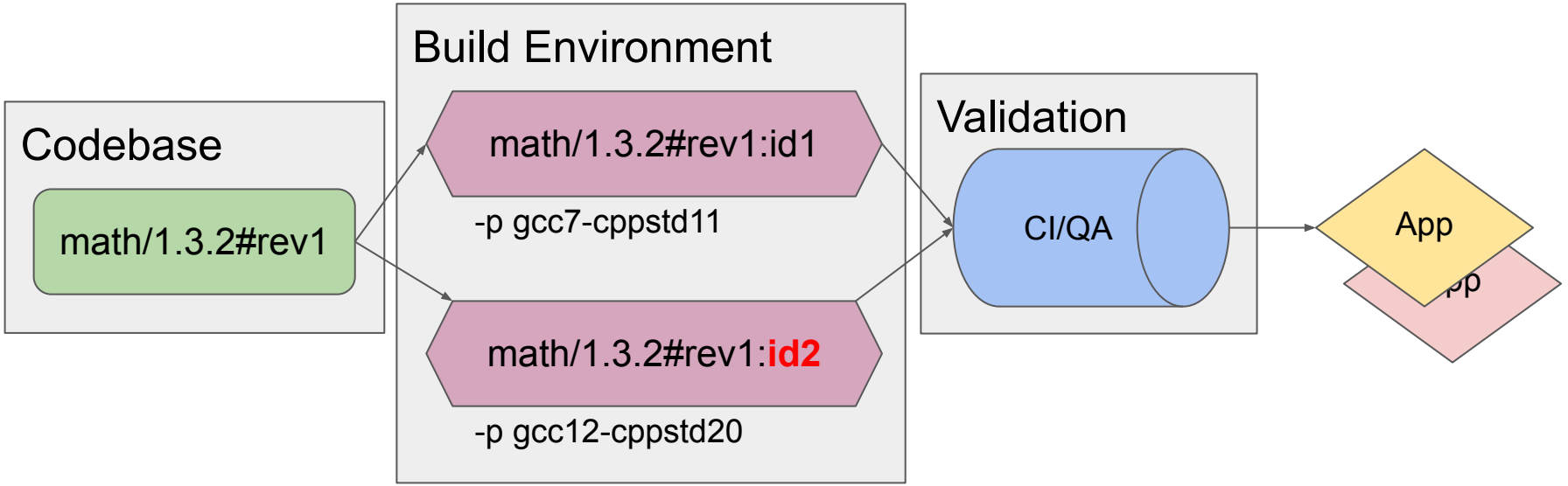
- Most C++ developers are still runnings C++11- but in a perfect world who doesn't want to update?

Today (build and shipped to customer)



A common solution

Future (Current development being validated along side LTS)



What’s new in Conan 2.0

Everything is new!

1.0

5 years, without breaking

60% new code, 20% backports

1.X ⇔ 2.0 compatible syntax subset

2.0

JFrog | CONAN C/C++ Package Manager

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

Everything is on docs.conan.io

We have been listening to you.

Usage and Feedback

- ~800K downloads/month from PyPI
- Designated as PyPI critical project (1% of most downloaded in whole PyPI)

PyPI Stats

Search

All packages
Top packages

Track packages

conan

PyPI page
Home page
Author: JFrog LTD
License: MIT
Summary: Conan C/C++ package manager
Latest version: 2.0.0

Downloads last day: 15,503
Downloads last week: 217,579
Downloads last month: 840,656

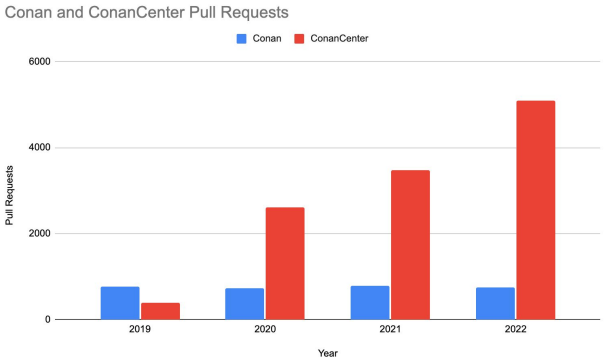
Analytics

Overview | Channels | Members

Data as of 02/16/2023, last updated 3 days ago

192 channels Export CSV Edit columns Last 30 Days Filter by channel name

Name	Created	Total membership	Messages posted	Members who posted	Members who viewed
# general	2016-08-16	22,292	3,097	107	739
# conan	2017-02-05	2,399	2,850	85	223
# boost	2016-09-02	2,874	5,087	54	178
# cmake	2017-06-21	3,751	1,018	49	230
# learn	2016-10-21	6,116	1,684	42	231
# off-topic	2017-11-17	919	2,707	31	75

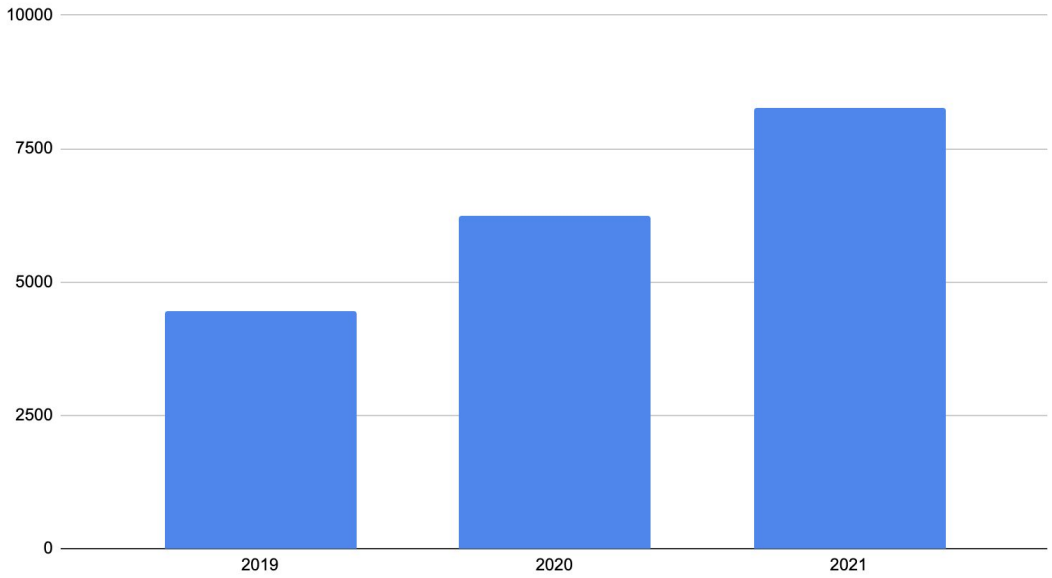


Support

+2000 Github issues / year

100 hr/year user video calls

Direct support (slack, almost daily)

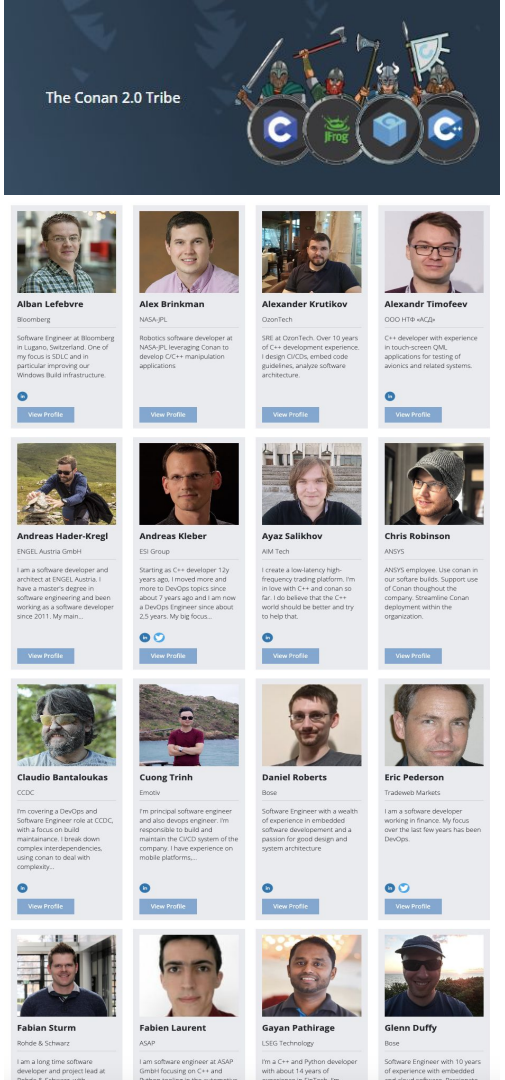


Artifactory servers running Conan in production and telemetry enabled (no firewalls)

JFrog | CONAN C/C++ Package Manager

Tribe 2.0 (conan.io/tribe.html)

- Bose
- TomTom
- Continental
- Nasa
- Apple
- Ansys
- Bloomberg
- Rohde & Schwarz
- Bosch
- ASAP
- Rti
- Zeiss
- Nasdaq
- Plex
- Keysight
- Datalogics
- VMWare
- … 50 more



Overview

- Some of the Feature Highlights:
  - Binary and Graph
  - Extensions and Commands
  - Reproducibility with Lockfiles
- Conclusions

1. Learning to Fly

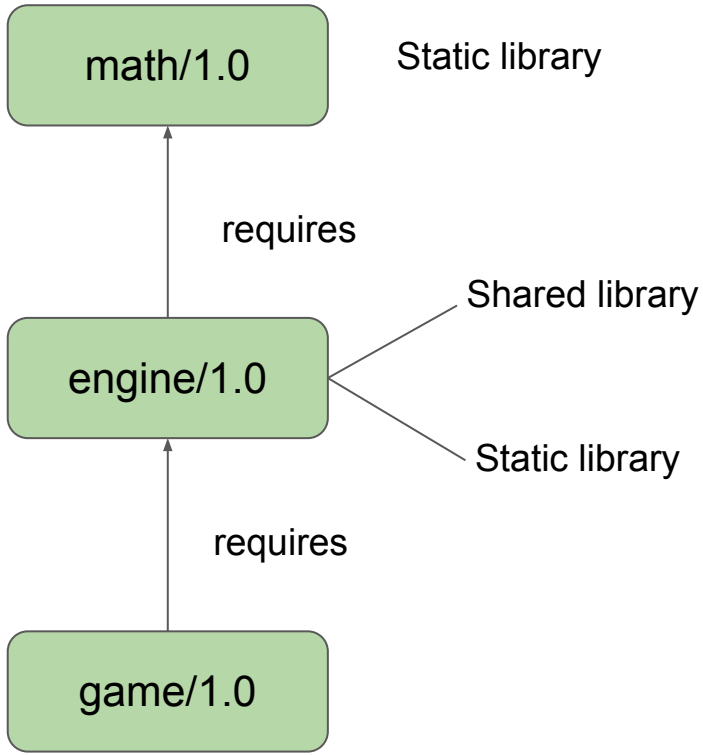
C and C++ are challenging

The first preview version of Conan was released in late 2016. Since then, Conan has experienced significant continued growth: empowering teams to bring their C++ projects into the DevOps era.

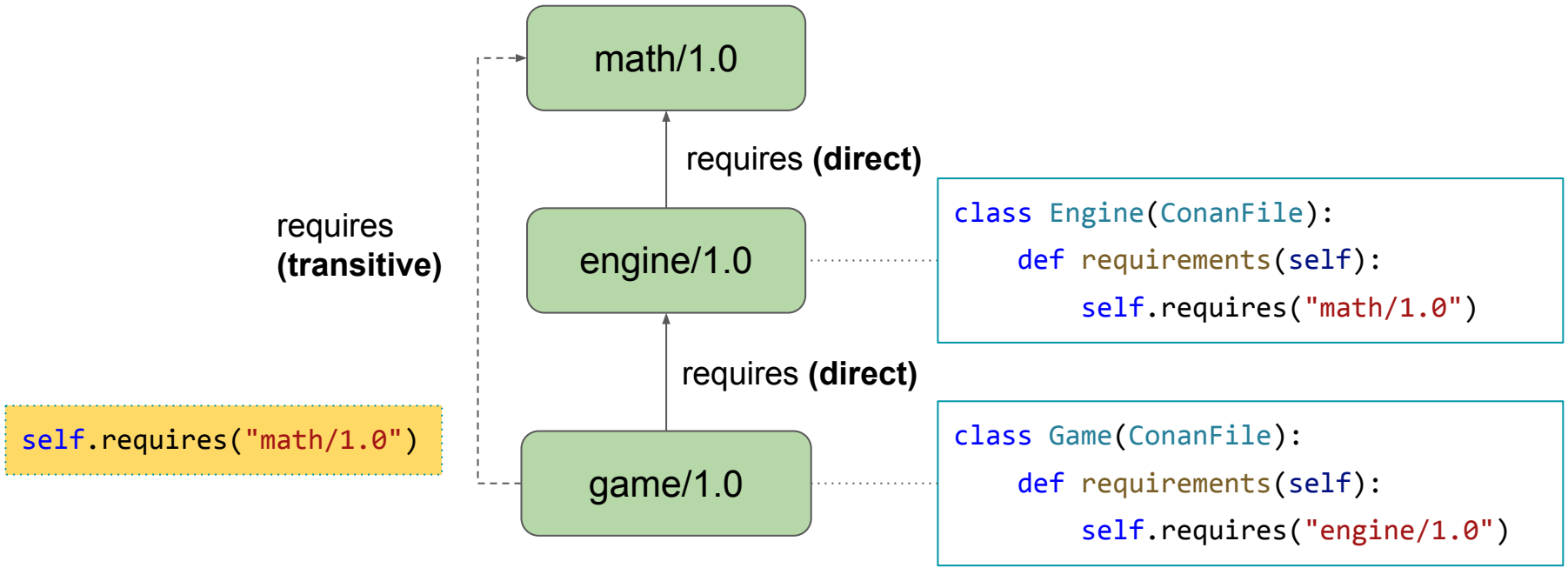
- **New documentation, written from scratch**
  - Hands-on tutorial covering the most common use cases
  - Comprehensive and better organized API reference
  - Newly structured examples section
  - Documented way to deploy Conan without Python in the developers computers

Modeling of “build-time only” Dependencies

Inspired by how build systems model transitive usage requirements, we created a model in which “requires” can be qualified with several different “traits”, that define how packages are required



Support Direct vs. Transitive Dependencies



Dependency Graph

- Correct linkage requirements
- Correct header visibility
- Possible hidden/private dependencies
- and many more (ACCU 2022)

Among different build systems!

Compatible “requires” syntax with 1.X

Conan 2.0 is way more efficient

Fetching only the binaries that it needs to build and run

- better `package_id` computation model)

Able to skip the download of large parts of the graphs binaries in many cases, accelerating install and build time.

- the new graph model
- the requirements traits and the new `package_type` recipe attribute

Will enable better modeling and management of dependencies.

2. Repeating yourself

C++ DevOps at Scale

So we are finally very excited to launch 2.0, with

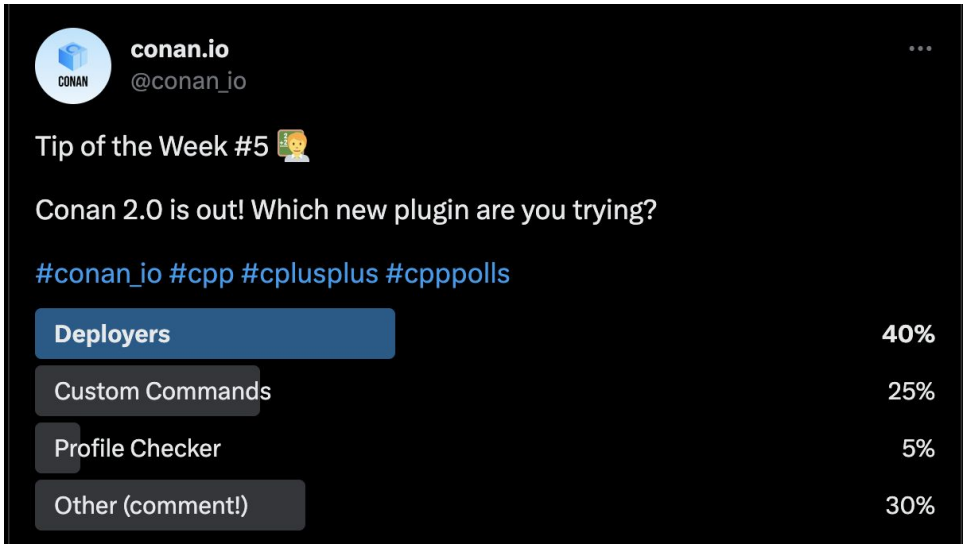
- streamlined and more powerful lockfiles
- new improved CI oriented tools
  - Introducing extensions
  - Custom commands and Python API

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.

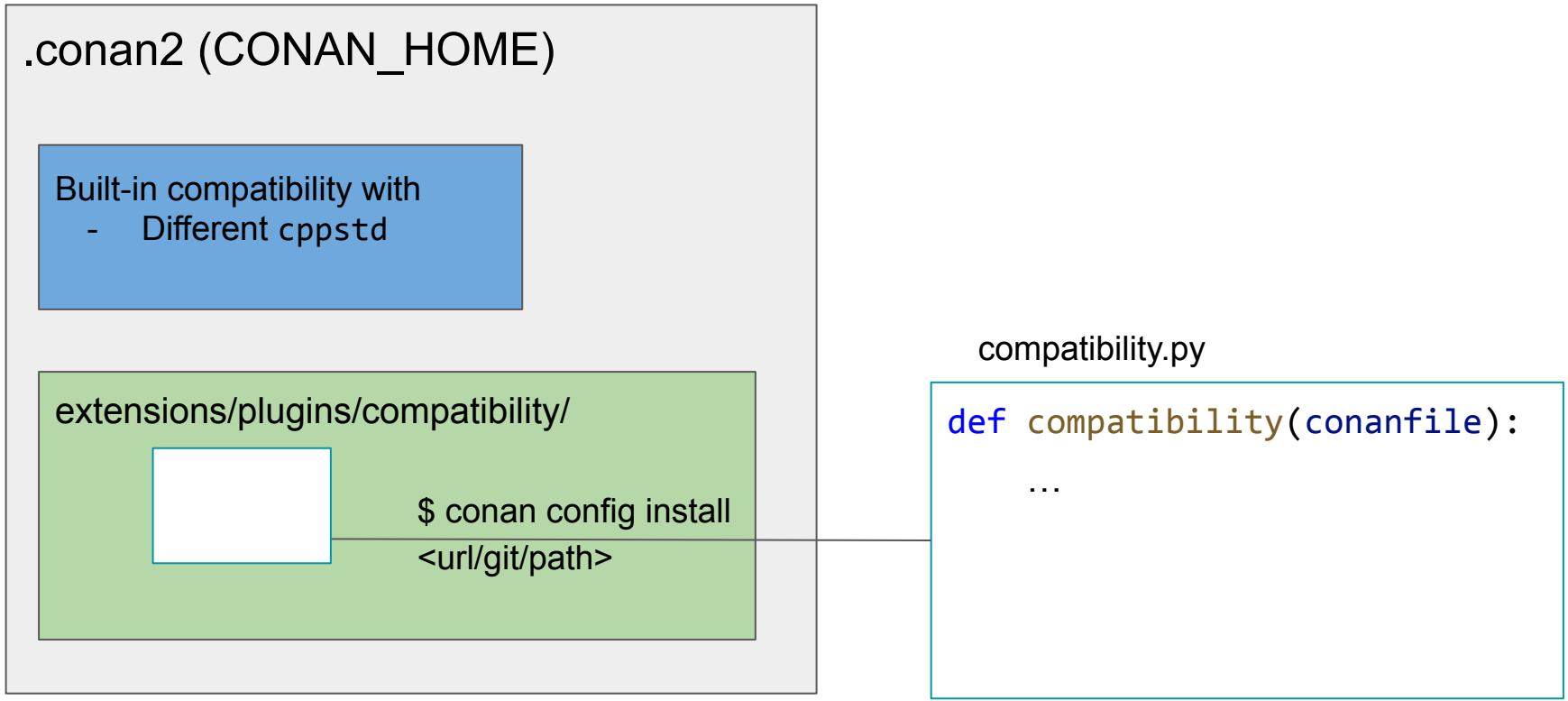
- Profile Checker
- Command Wrapper
- Package Signing



Deployers

- Flexible way to extract artifacts from cache
- Automate post-conan tasks
- Not in recipes, scale
- User customizable, “conan config install” installable

Compatibility Plugin



Python API and Custom Commands

- New Python API is released as “preview” no breaking changes
- Custom Commands built on top allow to tailor usage
- JSON output format to provide reliable stable output

Python API

```
# Conan 1.X API

class API:
    def install(path, profile_path, ...)
    def create(path, profile, ...)
    def lock(path, profile, ...)
    def search( ...)
    def remove(...)
    def upload(patterns, remote)
```

```
# Conan 2.0 API

class API:
    class RemoteAPI:
    class SearchAPI:
    class ProfileAPI:
        def get_profile(path, settings, options, conf)
    class GraphAPI:
        def graph(path, remotes, profiles, ...)
    class UploadAPI:
        def get_bundle(patterns, ..)
        def check_upstream(bundle, remote)
        def prepare(bundle)
        def check_integrity(bundle)
        def upload(bundle, remote)
```

Custom command

```python
import json

from conan.api.output import ConanOutput
from conan.cli.command import conan_command

def output_json(msg):
    return json.dumps({"greet": msg})


@conan_command(group="My own commands", formatters={"json": output_json})
def hello(conan_api, parser, *args):
    """
    Simple command to print "Hello World!" line
    """
    msg = "Hello World!"
    ConanOutput().info(msg)
    return msg
```

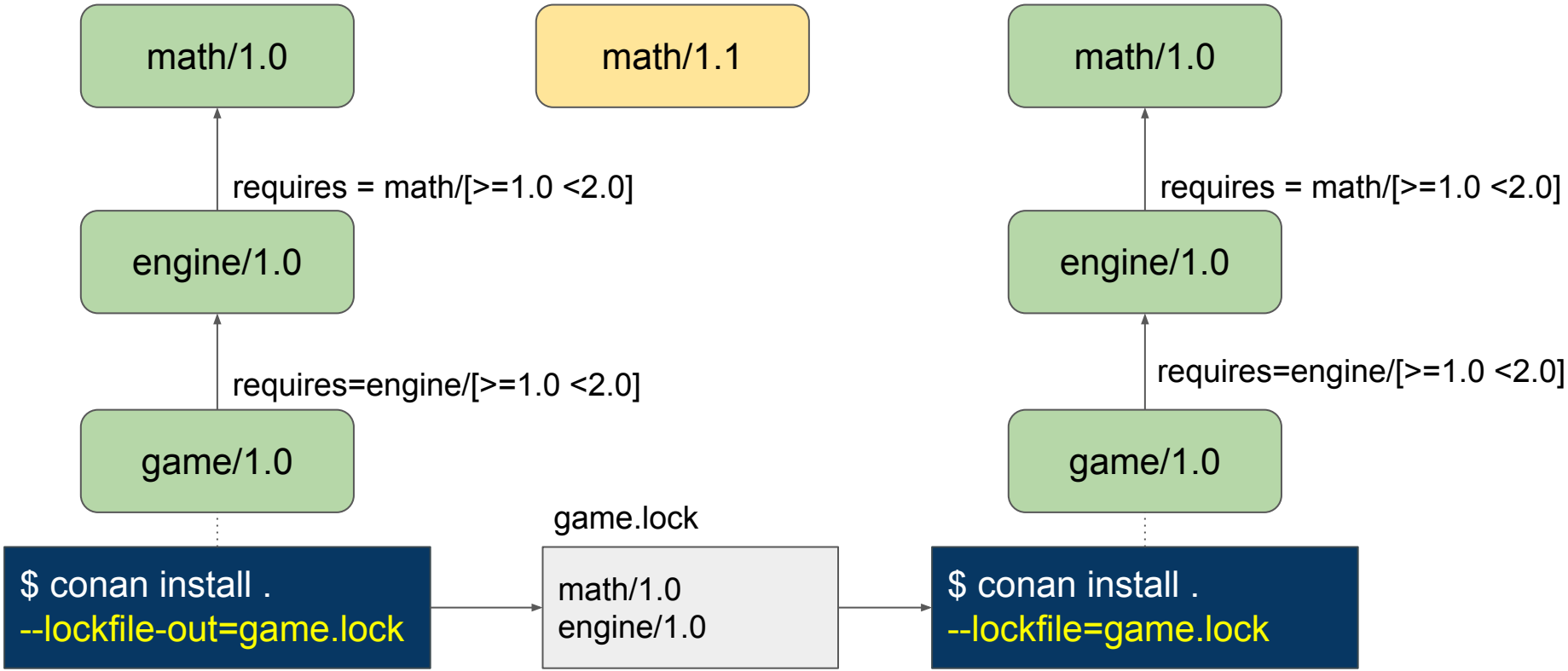
3. Building a dam

C and C++ are trusted by enterprises worldwide

Thousands of teams are using Conan in their C and C++ workflows, ranging from startups to many large enterprises. We have made every effort to keep our promise of stability, continually delivering updates to the Conan 1.x series without breaking changes.

“Toss it over to diego” <3

Reproducible dependencies: Lockfiles



Lockfiles 2.0

- One lockfile for all configurations
- Easily mutable
- Easily understandable
- Fully strict and partial modes
- Easily mergeable
- Manual commands to modify (override)
- Possible to use multi-project
- Code in codebase 10x shorter
- **Game changer for CI at scale**

Welcome Enterprise DevOps for C and C++

- Enterprise scale can be high
- Enterprise/domain requirements can be challenging
- Continuous Integration at scale is critical
- Thinking beyond package and dependency management
  - Programming over time => SW engineering (T. Winters)
  - Dependency and Package management over time => DevOps

Conclusions

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

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

Conclusion

pip install conan

https://conan.io

JFrog | CONAN C/C++ Package Manager