Conan - menadżer pakietów C/C++

Arkadiusz Jędrzejewski

3.12.2019

Co to?

- Menadżer pakietów
- Łatwiejsze zarządzanie zależnościami
- Lepsza reużywalność własnego kodu

Rozwiązania dla innych środowisk

- npm dla JavaScripta

- Cargo dla Rusta

- Nuget dla platformy .NET


python
TM

Po co? Obecne rozwiązania

- Ręczna kompilacja i konfiguracja
- Kopiowanie bibliotek lub kodu źródłowego
- Systemowe menadżery pakietów

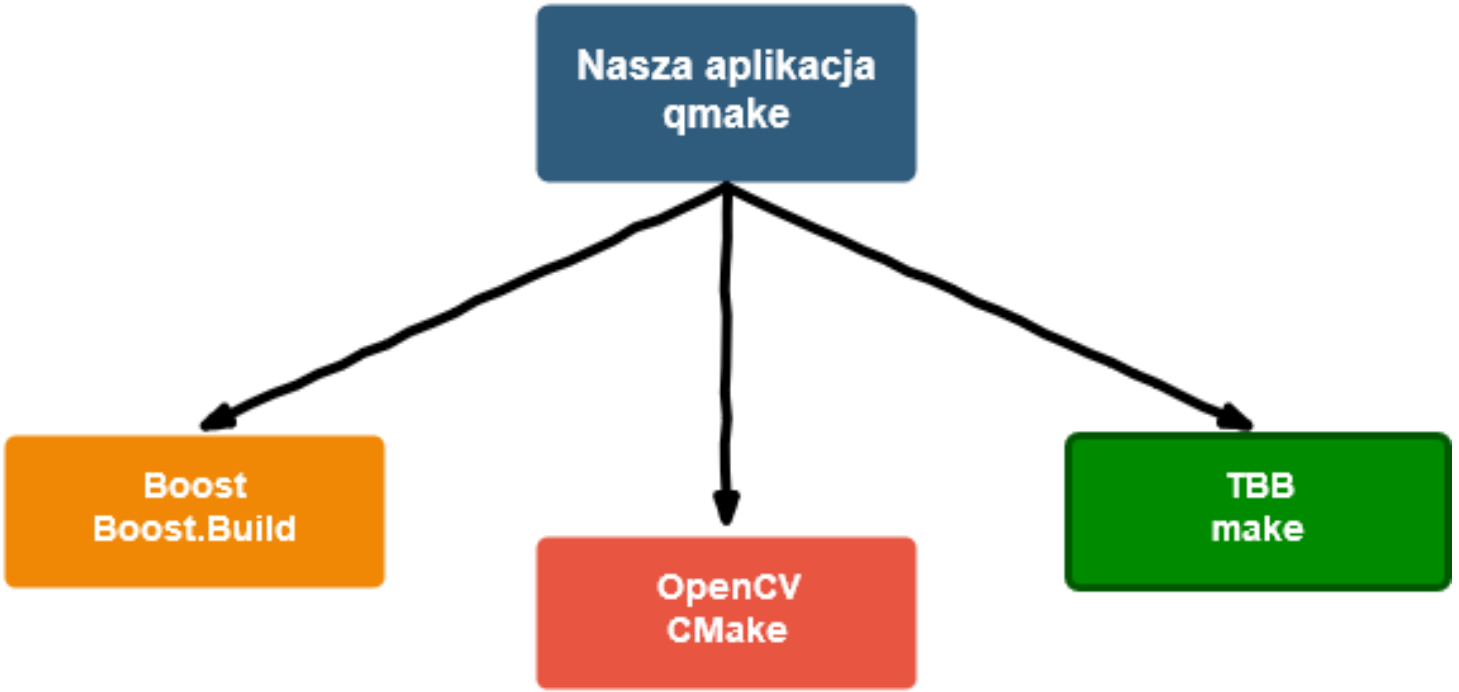
Po co?

- Łatwiejsza konfiguracja i zwiększona elastyczność
- Zwiększona dostępność bibliotek
- Brak problemu niekompatybilnego ABI

Rodzaje pakietów

- Oparte o prekompilowane biblioteki
- Oparte o kod źródłowy

Integracja wielu narzędzi



Dostępne repozytoria

- conan-center – domyślnie dostępne po instalacji
- public-conan – rozwijane przez społeczność

Instalacja

pip install conan

Interfejs w wierszu poleceń

```
Consumer commands
  install     Installs the requirements specified in a recipe (conanfile.py or conanfile.txt).
  config      Manages Conan configuration.
  get         Gets a file or list a directory of a given reference or package.
  info        Gets information about the dependency graph of a recipe.
  search      Searches package recipes and binaries in the local cache or in a remote.
Creator commands
  new         Creates a new package recipe template with a 'conanfile.py' and optionally,
              'test_package' testing files.
  create      Builds a binary package for a recipe (conanfile.py).
  upload      Uploads a recipe and binary packages to a remote.
  export      Copies the recipe (conanfile.py & associated files) to your local cache.
  export-pkg  Exports a recipe, then creates a package from local source and build folders.
  test        Tests a package consuming it from a conanfile.py with a test() method.
Package development commands
  source      Calls your local conanfile.py 'source()' method.
  build       Calls your local conanfile.py 'build()' method.
  package     Calls your local conanfile.py 'package()' method.
  editable    Manages editable packages (package that resides in the user workspace, but are
              consumed as if they were in the cache).
  workspace   Manages a workspace (a set of packages consumed from the user workspace that
              belongs to the same project).
Misc commands
  profile     Lists profiles in the '.conan/profiles' folder, or shows profile details.
  remote      Manages the remote list and the package recipes associated to a remote.
  user        Authenticates against a remote with user/pass, caching the auth token.
  imports     Calls your local conanfile.py or conanfile.txt 'imports' method.
  copy        Copies conan recipes and packages to another user/channel.
  remove      Removes packages or binaries matching pattern from local cache or remote.
  alias       Creates and exports an 'alias package recipe'.
  download    Downloads recipe and binaries to the local cache, without using settings.
  inspect     Displays conanfile attributes, like name, version and options. Works locally, in
              local cache and remote.
  help        Shows help for a specific command.
  graph       Generates and manipulates lock files.
```

Konfiguracja

- „conan remote” - zarządzanie repozytorium
- „conan profile” - zarządzanie profilami

Konwencja nazewnictwa pakietu

- [nazwa pakietu]/[wersja]@[właściciel]/[kanał]

  Przykład:

- catch2/2.5.0@bincrafters/stable

Conan dla odbiorcy – conanfile.txt

```
[requires]
opencv/4.1.0@conan/stable
boost/1.69.0@conan/stable
[generators]
cmake
[options]
opencv:tiff=False
opencv:nonfree=True
boost:shared=True
[imports]
bin, *.dll -> ./bin
lib, *.dylib* -> ./bin
```

Conan dla odbiorcy – conanfile.txt

```
conan install ../src/ --build missing
```

Conan dla odbiorcy – conanfile.txt

```
include(${CMAKE_BINARY_DIR}/conanbuildinfo.cmake)
conan_basic_setup()
```

Conan dla twórcy pakietu – conanfile.py

```python
class ImageConan(ConanFile):
    name = "image"
    version = "2019.1.6"
    license = "Optinav sp. z o.o. 2019"
    author = "Optinav"
    url = "docs placeholder"
    description = "Image container library."
    topics = ("image", "container", "c++11")
    settings = "os", "compiler", "build_type", "arch"
    options = {"shared": [True, False]}
    default_options = "shared=False"
    generators = "cmake"

    def source(self):···

    def build(self):···

    def requirements(self):···

    def package(self):···

    def package_info(self):···
```

Conan dla twórcy pakietu – conanfile.py

```
conan new -st image/2019.1.6@optinav/testing
```

Conan dla twórcy pakietu – conanfile.py

```
conan create . image/2019.1.6@optinav/testing
```

Dziękuję za uwagę!