CMake course for Elbit-Elisra

Alex Kushnir

CLARITUNE
Making Things Happen

Summary of the first day

What CMake is?

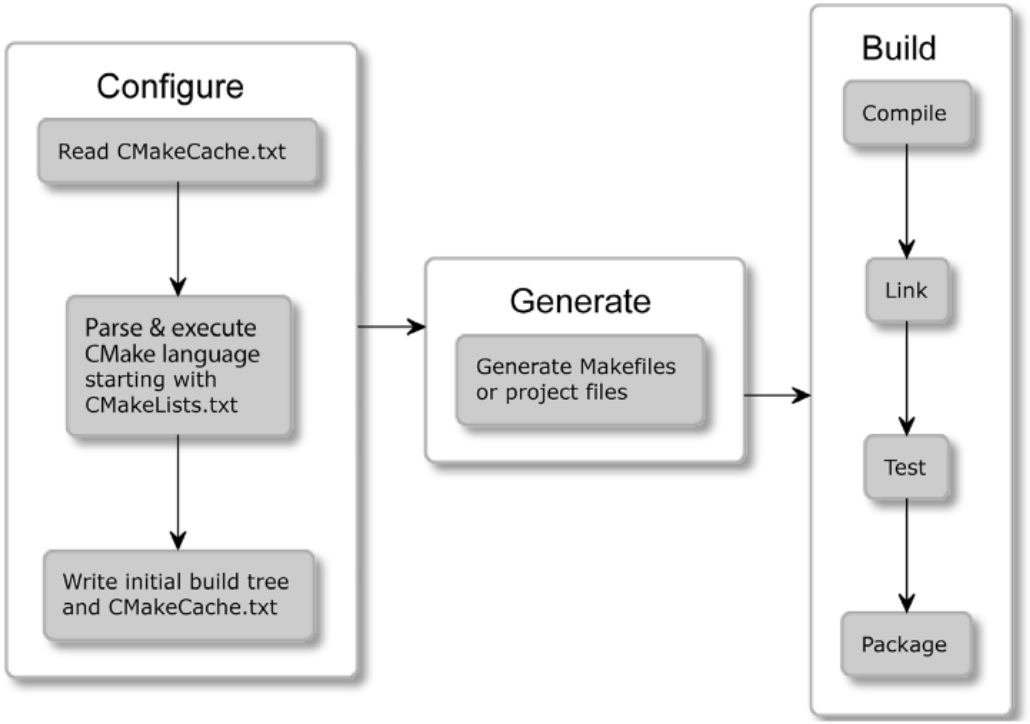
- A family of tools designed to build, test and package software.
- Cross-platform (Windows/Linux/macOS/Cygwin).
- Most modern compilers and toolchains are supported.
- Able to generate project files for all popular IDEs (Visual Studio, Eclipse, CLion, etc.)
- Able to generate Makefiles and ninjafiles
- Rich ecosystem.
- Old features get deprecated to keep CMake lean.
- Developed and maintained by Kitware Inc.

Motivation

- Managing artifacts creation is complicated
- Dependent on environment
- Abstraction hiding complexities
- We never want to “reinvent the wheel”
- Can also fetch, test and install

How it works?

- The process of building has 3 stages
  - Configuration
  - Generation
  - Building



Basic Example

```
cmake_minimum_required(VERSION 3.28)
set(CMAKE_CXX_STANDARD 23)

project(Calculator)

add_executable(calc Main.cpp Calculator.cpp)
```

Glossary

- **Source tree**
- **Build tree**
- **Project file**
- **Cache file**
- **Generated files**
- **Preset files**
- **Package definition file**

Topics Brief

- CMake Commands (slides 29 – 38)
- Variables and Lists (slides 39 – 51)
- Scripting: Conditions (slides 52 – 57)
- Scripting: Loops (slides 58 – 62)
- Scripting: Functions and Macros (slides 63 – 73)
- Partitioning the Project (slides 78 – 83)
- Using the Project Environment (slides 84 – 86)
- Configuring the Toolchain (slides 87 – 95)
- CMake Targets (slides 98 – 109)
- Compilation and Linking (slides 110 – 155)
  - Compiler options configuration (slides 120 - 121)
  - Adding source files to targets (slides 122 – 127)
  - Preprocessor configuration (slide 128)
  - Shared and Static Libraries (slides 147 – 149)

Today!

- Warm-up and alignment
- Managing dependencies with CMake
- Writing CMake presets
- Testing with CMake

Additional Completion

- Program-analysis tools
- Installation and packaging
- Your questions!