

Inhaltsverzeichnis

Vorwort	v
1. Einleitung – Was ist CMake?	1
1.1. CMake – Ein Build-System-Generator	1
1.2. Zusätzliche Informationen – Was Sie vor dem Lesen wissen sollten . .	3
2. Grundlagen – Bevor es richtig los geht	11
2.1. Einrichtung des Systems – Installation der wichtigsten Komponenten .	11
2.2. Der CMake-Build-Prozess – So läuft das also	15
3. Der Einstieg in CMake – Jetzt gehts los	29
3.1. Die erste CMakeLists.txt – Es reichen drei Befehle	29
3.2. Kommentare – Auch in CMake wichtig	33
3.3. Die drei Befehle – Mehr Optionen	35
3.4. CMake-Entwicklung – Targetfokussierung Teil I	41
4. Klassisches Programmieren – Viele Befehle gibt es auch in CMake	43
4.1. Variablen – Veränderlicher Wertespeicher	44
4.2. Properties – So ähnlich wie Variablen	62
4.3. Verzweigungen - Befehle ausführen oder doch lieber nicht?	71
4.4. Schleifen – Und noch mal und noch mal und	80
4.5. Funktionen und Makros – Befehle selbst schreiben	87
5. Bibliotheken – Zusammenhängenden Programmcode organisieren	105
5.1. Erstellen – Bibliotheken definieren	106
5.2. Header-Dateien – Einbinden aus einem separaten Ordner	111
5.3. Verlinken – Beziehung zwischen einer Bibliothek und einem Target . .	115

6. Build-Konfigurationen – Was bauen wir denn?	125
6.1. Wahl der Build-Konfiguration – Was bringt das?	125
6.2. Generator Expressions – Entscheidungen im Generierungsschritt . . .	131
6.3. Single- und Multikonfigurationsgeneratoren – Worauf man achten sollte	143
7. Projektstrukturierung – Wie Sie Ihr Projekt aufteilen können	147
7.1. Ordner und Dateien einbinden – Wenn es komplexer wird	147
7.2. Projektstrukturierung – Ein Vorschlag	162
7.3. CMake-Scriptstrukturierung – Ein weiterer Vorschlag	164
8. Linker- und Compiler-Optionen – Wie CMake funktioniert	167
8.1. Compiler-Flags – Kommunikation mit dem Compiler	167
8.2. Linker-Flags – Was beim Linken passiert	177
8.3. CMake-Entwicklung – Targetfokussierung Teil II	179
9. Modules und Packages – Code im Paket	181
9.1. Modules – Vorgefertigter CMake-Code	181
9.2. Packages – Ein ganzes Code Paket	192
10. Testing – Testen des Projektcodes	223
10.1. Definieren und Ausführen – Los gehts!	223
10.2. Kriterien - Spezifizieren wann Tests erfolgreich sind und wann nicht .	229
10.3. Labeln – Tests zusammenfassen	233
10.4. Testbibliotheken – GoogleTest und Catch2	237
11. Nützliche Features – Was gibt es sonst noch?	249
11.1. Doxygen – Ein Dokumentationstarget definieren	249
11.2. Clang-Tidy – Automatische Codeüberprüfung	254
11.3. FetchContent – Automatisiert Code herunterladen	258
11.4. Debugging – CMake-Code überprüfen	262
Schlusswort	265
Anhang	267
A. DLL Export unter Windows	267
B. Regular Expressions	269