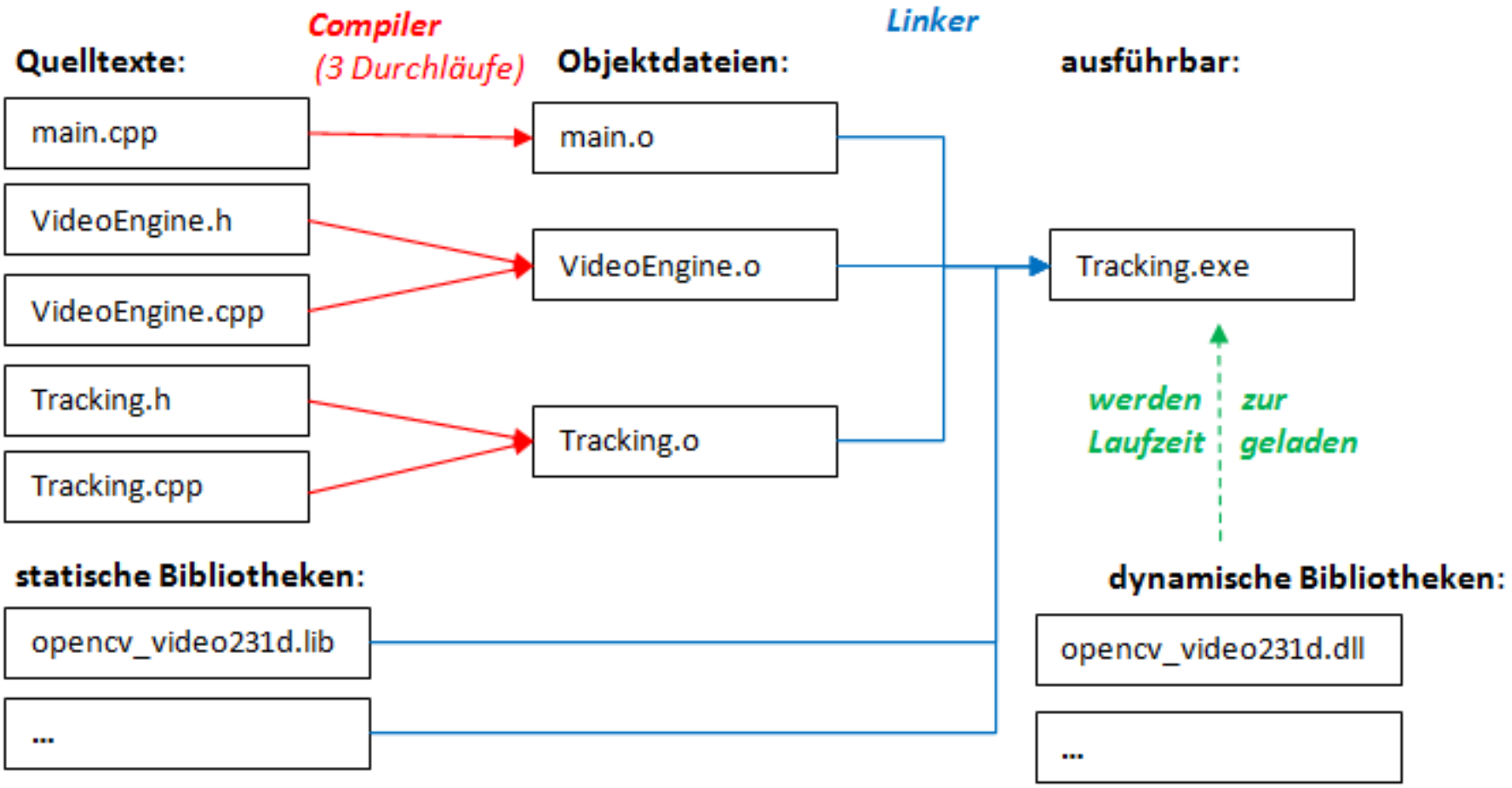
C++ Build Tools



Elementare Datentypen

Datentyp	Wortbreite	kleinste Zahl	größte Zahl
`char`	8	-128	127
`short`	16	-32768	32767
`int`	32	-2^{31} = -2147483648	$2^{31}-1$ = 2147483647
`long`	32	-2^{31} = -2147483648	$2^{31}-1$ = 2147483647
`unsigned char`	8	0	255
`unsigned short`	16	0	65535
`unsigned int`	32	0	2^{32-1} = 4294967295

Unterschiede zu Java

- **bool** für Wahrheitswerte, mit den Werten false, true
- **char** meistens 8-Bit, d.h. ASCII Zeichensatz; Alternative (compiler-abhängig): **wchar_t**
- **char** wird oft auch als 8-Bit Zahl verwendet
- **long** häufig 32-Bit; Alternative (compiler-abhängig): **__int64** oder **long long**
- **signed/unsigned** für Ganzzahl Datentypen gibt es die Varianten **signed/unsigned** (wird vorangestellt)

Beispiel:

```
unsigned char colorChannel;
```

typedef

Eine Art „Alias für Datentypen“;

Beispiel: Definition eines Datentyps **byte**

```
typedef unsigned char byte;
byte colorChannel;
```

Typumwandlungen

- C++ wandelt Werte fast immer in den passenden Datentyp um, d.h. keine Compiler-Überprüfung

Beispiel:

```
double x = 6.5;
int i;
i = x;  // Umwandlung double->int
```

Nicht-Initialisierung von Variablen

In C++ werden Variablen nicht automatisch initialisiert; der Compiler zeigt keinen Fehler.

Kontrollstrukturen - Achtung

Nicht empfehlenswert:

```
if (x = 0) {…}
```

(Bedingung wird nie ausgeführt)

```
if (x) {…}
```

Besser:

```
if (x == 0) {…}
```

```
if (x != 0) {…}
```

Präprozessor-Anweisungen

- **#include**

  #include "lokaleDatei.h"

  #include "SystemDatei.h"

- **#define**

  #define DATEI_H

  #define STEREO 2

- **#ifdef, #else, #endif**

  #ifdef WIN32

  …

  #endif

Strings

- C: char* cstring= "Hello";
- C++: std::string stdstring= "Hello";
- Qt: QString qstring = "Hello";

Umwandlungen:

stdstring.c_str(): C++-String→ C-String

qstring.toStdString(): Qt-String → C++ String