Praktikum Informatik 1

Installation der Arbeitsumgebung

Sommersemester
2025

Version vom 17. April

Prof. Dr.-Ing. Jürgen Roßmann
Institut für Mensch-Maschine-Interaktion
Im Süsterfeld 9, 52072 Aachen

Inhaltsverzeichnis

1 Installation der Entwicklungsumgebungen für Windows

Dieses Kapitel liefert eine kurze Anleitung, um alle für dieses Praktikum benötigten Programme unter Microsoft Windows 10/11 korrekt zu installieren und einzurichten. Die hier aufgeführten Installationsschritte sind nur nötig, wenn Sie eine funktionierende Arbeitsumgebung auf Ihrem Rechner zu Hause oder Ihrem Laptop einrichten möchten. Sie sind allerdings nicht verpflichtet, die Software auf Ihrem privaten Rechner zu installieren. Auf den Rechnern des CIP-Pools stehen Ihnen die genannten Programme zur Verfügung und eine Installation ist nicht mehr nötig. Unabhängig davon, ob Sie die vorinstallierten programme auf den CIP-Pool-Rechnern oder selber installierte Programme auf Ihrem eigenen Rechner, sollten Sie sich mit den Tipps ab Seite 11 beschäftigen, da Sie hierdurch in Ihrer Arbeit unterstützt werden.

Hinweis: Eine kostenlose Windows-Lizenz ist über *Azure Dev Tools for Teaching* erhältlich. Eine Anleitung finden Sie auf der Seite des IT-Centers unter *Beschaffung & Software* → *Microsoft Produkte* → *Verträge mit Microsoft* → *Azure Dev Tools for Teaching*. Das IT-Center der RWTH-Aachen steht Ihnen für weitere Fragen diesbezüglich zur Verfügung. Andere Betriebssysteme (Unix/ Linux, Mac OS, usw.) werden nicht unterstützt.

Die benötigte Software zur Einrichtung der Entwicklungsumgebungen liegt auf dem *Sciebo-Server* der RWTH. Den zugehörigen Link finden Sie im *Moodle* in der Rubrik *Hyperlinks*. Klicken Sie auf den Eintrag *Software Praktikum Informatik 1 - 2025*, um das Archiv herunter zu laden. Das Archiv ist ca. 1.8GB groß. Im CIP-Pool kann das Laden teilweise Stunden dauern, da Sie sich die WLAN-Bandbreite mit allen anderen teilen müssen.

Eine allgemeine Beschreibung bzgl. der Entwicklungsumgebungen *Eclipse* und *Qt-Creator* sowie eine detaillierte Beschreibung bzgl. deren Handhabung finden Sie im Skript zum Praktikum.

Installieren Sie die Software unbedingt in der hier vorgegebenen Reihefolge, damit Sie nicht nachträglich von Hand diverse Einstellungen ändern müssen.

1.1 Java-Laufzeitumgebung [03:35]

In diesem Praktikum wird *Eclipse* zusammen mit einer Erweiterung zum Entwickeln von C und C++ -Programmen, genannt *CDT*, als Entwicklungsumgebung verwendet. Da Eclipse ein Java-Programm ist, benötigen Sie zunächst eine Java-Laufzeitumgebung.

1. Um zu testen, ob Java schon installiert ist, öffnen Sie die Eingabeaufforderung, indem Sie in der Taskleiste bei *Suchen* das Kommando *cmd.exe* eingeben, dann *Enter*. Geben Sie in dem geöffneten Fenster *java -version* ein. Wenn es eine Fehlermeldung gibt, das die Datei nicht gefunden wurde, ist Java nicht installiert. Kommt irgendwas mit Version xy größer gleich 17, dann ist eine passende Java-Version schon installiert und Sie müssen nichts machen. Wichtig: Eclipse benötigt mindestens Version 17.
2. Ansonsten installieren Sie Java, indem Sie die Datei aus dem Sciebo-Ordner nach C:\extrahieren. Sehen Sie nach dem Auspacken nach, ob auf Laufwerk C:\ ein Ordner *jdk-23* erstellt worden ist bzw. dass Sie das Verzeichnis C:\jdk-23:\bin erreichen können.

3. Damit Windows die Dateien auch findet, muss die Systemumgebungsvariable *Path* angepasst werden. Das Vorgehen ist in Abbildung 1.1 zu sehen: Geben Sie hierzu in das Suchfeld in der Taskleiste solange *Systemumgebungsvariablen* ein, bis *Systemumgebungsvariablen bearbeiten* erscheint. Klicken Sie in dem sich öffnenden Fenster auf *Umgebungsvariablen*, markieren Sie im unteren Fenster *Path* und klicken Sie darunter auf *Bearbeiten*. Im nächsten Fenster klicken Sie auf *Neu* und geben in der neuen Zeile C:\jdk-23\bin ein, dann *Enter, ok, ok, ok*.
4. Wenn Sie jetzt in der Eingabeaufforderung, die Sie neu starten müssen, noch mal *java -version* eingeben, sollte es keine Fehlermeldung mehr geben. Damit ist Java installiert.

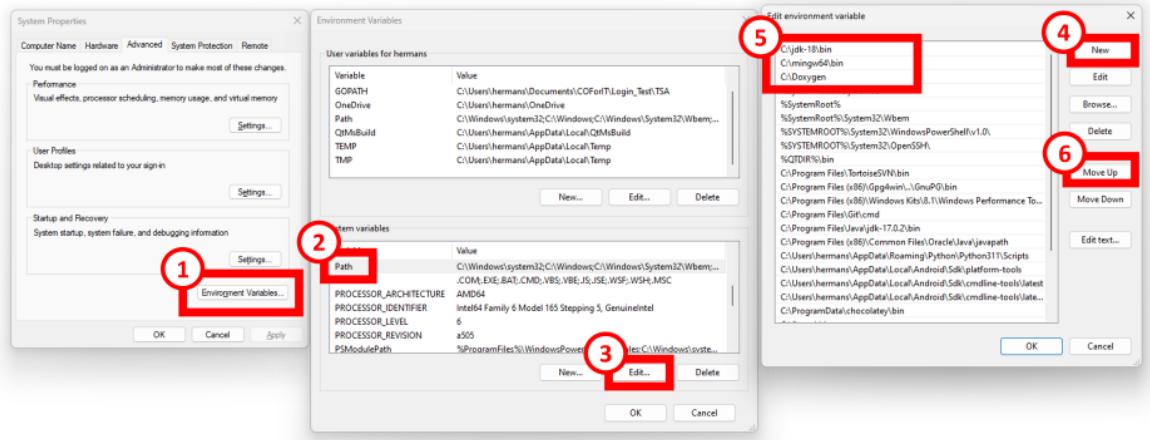

Abbildung 1.1: Setzen der Umgebungsvariablen: 1. Umgebungsvariablen bearbeiten 2. In Systemvariablen PATH auswählen 3. Bearbeiten 4. Jeweils drei neue Einträge erstellen 5. Entsprechende Werte eintragen 6. Einträge nach oben schieben

1.2 MinGW-w64 [07:55]

In diesem Praktikum soll auf dem Betriebssystem *Windows* gearbeitet und Programme erstellt werden, welche auf demselben lauffähig sind. Hierzu eignet sich z.B. der Compiler *MinGW-w64*, welchen Sie im Rahmen dieses Praktikums nutzen werden. Gehen Sie bei der Installation des Compilers genauso vor wie bei der Installation von Java. Entpacken Sie das Archiv *mingw64.zip* wieder nach C:\. Passen Sie wieder die Systemumgebungsvariable *Path* wie in Abbildung 1.1 an, geben Sie aber jetzt als Wert C:\mingw64\bin an. Damit ist der Compiler installiert.

1.3 Eclipse [09:38]

Gehen Sie bei der Installation von Eclipse ebenso vor wie bei den Vorherigen. Entpacken Sie das Archiv *eclipse.zip* nach C:\. Passen Sie wieder die Systemumgebungsvariable *Path* wie in Abbildung 1.1 an, *Eclipse* werden Sie normalerweise über den Explorer starten. Es ist aber praktisch, sich eine Verknüpfung auf dem Desktop oder direkt in der Taskleiste zu erstellen. In dem Ordner C:\eclipse klicken Sie mit der rechten Maustaste auf *eclipse* (mit dem runden, blauen Symbol) und dann entweder auf *An Start anheften*, → *Senden an* → *Desktop* oder *An Taskbar anheften*. Damit ist *Eclipse* installiert.

Beim ersten Starten von Eclipse werden Sie nach einem Ort für den „Workspace“ gefragt (siehe Abbildung 1.2). Ein Workspace ist ein Ordner, in dem Eclipse Ihre Projekte und die dazugehörigen *Metadaten* speichert. Wählen Sie einen Ort für den Workspace. In dem Pfadnamen dürfen keine Leer- oder Sonderzeichen vorkommen. Bestätigen Sie den Dialog.

Hinweis: Erzeugen Sie Ihren Workspace-Ordner im CIP-Pool auf dem Laufwerk *U:\workspace*. Achten Sie auch diesmal darauf, dass der Pfad keine Leer- und Sonderzeichen enthält.

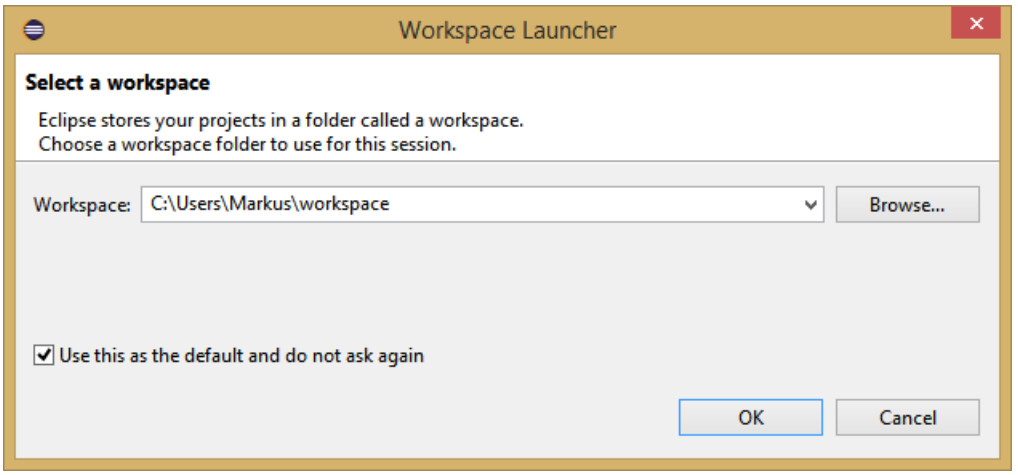

Abbildung 1.2: Eclipse fragt beim Start nach dem Ort für den Workspace.

1.4 Doxygen [27:17]

Doxygen erlaubt es, aus Kommentaren im Quelltext automatisch eine Dokumentation zu erzeugen. Die grundlegenden Befehle von Doxygen werden Sie während der Bearbeitung dieses Praktikums erlernen. Wie beim Compiler auch, handelt es sich hierbei um ein externes Tool, das separat installiert werden muss.

Gehen Sie bei der Installation wie bei den vorherigen Installationen vor.

1. Entpacken Sie das Archiv *Doxygen.zip* wieder nach C:\.
2. Ergänzen Sie in den Systemumgebungsvariablen die Variable *Path* um den Wert C:\Doxygen (ohne bin). Bestätigen Sie alles mit *ok*.
3. Geben Sie in einer neuen Eingabeaufforderung *doxywizard* ein, nicht *doxygen*. Es öffnet sich ein Fenster, das *Doxygen GUI frontend*. Schliessen Sie das Fenster wieder, fertig.

1.4.1 Eclox Doxygen Plugin for Eclipse [29:00]

Damit die Optionen zum Erstellen eiens Doxyfiles und Kompilieren der Dokumentationsdateien in Eclipse angezeigt wird, muss das *Eclox Plugin* installiert werden. Gehen Sie dabei wie folgt vor:

1. Öffnen Sie das Menü *Help* → *Eclipse Marketplace*.
2. In dem sich öffnenden Fenster geben Sie unter *Find* den Namen des Plugin *Eclox* ein und bestätigen Sie mit Enter.
3. Es sollte *eclox 0.13.0* angezeigt werden. Klicken Sie auf *Install*.

4. Entfernen Sie das Häckchen bei *Doxygen binaries*, denn Sie haben im vorherigen Abschnitt Doxygen bereits installiert. Klicken Sie auf *Confirm*, akzeptieren Sie die Lizenzbedingungen und klicken Sie auf *Finish*.

Starten Sie Eclipse neu. Sie sollten nun in der Symbolleiste ein blaues @-Zeichen sehen.

1.5 Git [36:55]

Das Versionskontrollssystem Git hilft Ihnen, Änderungen in Ihrem Code auch noch nach einiger Zeit nachzuvollziehen und bietet außerdem die Möglichkeit verschiedene Fortschrittsstände effizient zu sichern und mit anderen Geräten zu synchronisieren. Eine ausführliche Anleitung finden Sie im Skript zwischen Versuch 5 und 6.

Git kann von `https://git-scm.com/` installiert werden. Den Installer finden Sie auch im Sciebo-Ordner untere Software. Öffnen Sie diesen und klicken sich durch das Installationsmenü, wobei in der Regel die Default-Einstellungen übernommen werden:

1. Bei der Auswahl der zu installierenden Komponenten können Sie die Default-Einstellung beibehalten. Ebenso beim Default Editor.
2. Im nächsten Schritt beim Default Branch Name klicken Sie auf *Override* und tragen *main* ein.
3. Im nächsten Schritt werden Sie nach dem Hinzufügen von Git zu *PATH* gefragt. Wählen Sie *Git from command line and also from 3rd-party software*, um Git automatisch zu *PATH* hinzufügen.
4. In den restlichen Schritten aktzeptieren Sie die vorausgewählte Einstellung und klicken abschließend auf Install.

1.5.1 EGit Plugin for Eclipse [39:25]

Ähnlich für Doxygen muss für die Nutzung in der Eclipse IDE ein Plugin installiert werden. In den neueren Versionen von Eclipse ist dieses zumeist schon standardmäßig installiert. Gehen Sie dabei wie folgt vor:

1. Öffnen Sie wieder das Menü *Help* → *Eclipse Marketplace*.
2. In dem sich öffnenden Fenster geben Sie unter *Find* den Namen des Plugin *EGit* ein und bestätigen Sie mit Enter.
3. Es sollte *EGit - Git Integration for Eclipse 6.7.0* oder eine neuere Version angezeigt werden. Klicken Sie auf *Install*.

Starten Sie Eclipse neu. Wenn Sie nun *Window* → *Preferences* sollten dort eine neue Menüoption *Version Control (Team)* → *Git* sehen.

1.6 Qt-Creator [43:05]

Qt ist eine C++ Klassenbibliothek, die es mit relativ einfachen Mitteln erlaubt, grafische Benutzeroberflächen für C++ zu gestalten. Bei Verwendung der Qt-Bibliotheken der Version 6.x sind die selbst programmierten Projekte lauffähig auf Linux, MacOS und auch Windows, ohne Änderungen am Quellcode vornehmen zu müssen.

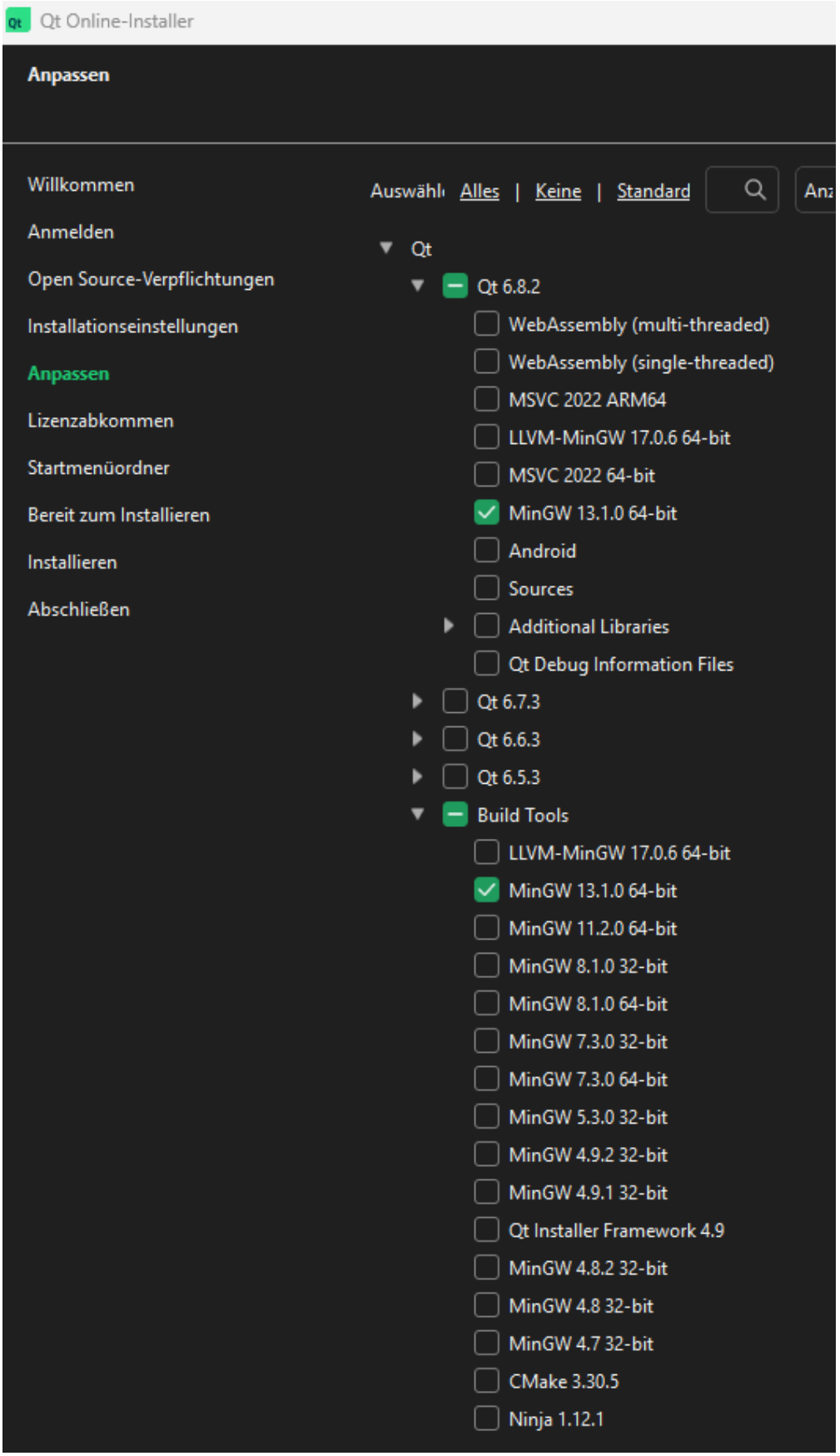

Abbildung 1.3: Auszuwählende Komponenten im Qt Installer. Die Versionsnummern könnten bei neueren Versionen anders aussehen

Der Qt Creator ist eine C++/Qt-Entwicklungsumgebung (IDE), die besonders für die Entwicklung von plattformunabhängigen C++ -Programmen und QML-Oberflächen mit der Qt-Bibliothek gedacht ist.

Starten Sie das Programm *qt-online-installer-windows-x64-4.9.0.exe* aus dem Softwareordner. Gehen Sie dann wie folgt vor:

1. Auf der Seite *Anmelden*, klicken Sie auf *Registrieren* und erstellen Sie einen Qt-Account.
2. Lesen Sie die Bedingungen zu Open Source-Verpflichtungen und wenn Sie diesen zustimmen, aktivieren Sie die beiden Häckchen.
3. Wählen Sie in den *Installationseinstellungen* das vorausgewählte Verzeichnis *C:\Qt*, sowie die vorausgewählte Einstellung *Qt 6.x für Desktop-Entwicklung*. Wählen Sie zusätzlich die Option *Benuterdefinierte Option*.
4. Auf der Seite *Anpassen* wählen Sie unter *Qt → Qt 6.x.y* die Option *Additional Libraries* ab. Und unter *Qt → Build Tools* klicken Sie die Option *MinGW 13.1.0 64-bit* entsprechend der obigen Auswahl. Die Optionen *CMake* und *Ninja* können Sie deaktivieren. Die Auswahl für die Qt-Version ist auch nochmal in Abbildung1.3 zu sehen.
5. Wenn Sie mit den *Lizenzabkommen* einverstanden sind bestätigen Sie diese.
6. Behalten Sie den Startmenüordner bei und klicken Sie abschließend auf *Installieren*. Die Gesamtgröße sollte etwa 2.7 GB betragen.

1.7 Software wieder deinstallieren

Qt und Git müssen in der Systemsteuerung unter *Programme und Features* explizit deinstalliert werden. Bei Java, Mingw-w64, Doxygen und Eclipse reicht es, einfach die Ordner unter C:\ wieder zu löschen.

2 Testen der Entwicklungsumgebungen

Bevor es nun mit dem Praktikum richtig losgeht, sollten Sie zuerst testen, ob die Entwicklungsumgebung im CIP-Pool oder zu Hause richtig eingerichtet und konfiguriert ist. Dazu gibt es Testprogramme, welche alle benötigten Features enthalten. Sollten diese Programme an Ihrem Rechner im CIP-Pool nicht funktionieren, melden Sie sich bitte bei einem der Betreuer. Bei Problemen mit Privatrechnern versuchen wir im Rahmen unserer Möglichkeiten ebenfalls behilflich zu sein.

2.1 Test Eclipse [12:35]

Wenn bei der Installation keine Fehler aufgetreten sind, sollten Sie nun in der Lage sein, ein neues *Hello World C++ Project* mit der Toolchain *MinGW GCC* anzulegen.

- Starten Sie Eclipse.
- Wählen Sie *File* → *New* → *C/C++ Project* → *C++ Managed Build*.
- Geben Sie dem Projekt einen sinnvollen Namen, z.B. *hello_world*.
- Unter *Project type* wählen Sie *Hello World C++ Project*.
- Bei *Toolchains* muss *MinGW GCC* ausgewählt werden.
- Bestätigen Sie das Fenster mit *Finish*.
- Kompilieren Sie das Projekt über *Project* → *Build Project*.
- Führen Sie es über *Run* → *Run As* → *1 Local C/C++ Application* aus.

Wenn alles richtig installiert wurde, sollte nun in der Konsole *!!!Hello World!!!* stehen.

2.2 Test Doxygen [30:40]

Um das Testprogramm zu starten, gehen Sie wie folgt vor:

- Laden Sie die Vorlagen aus Moodle herunter und entpacken Sie diese in einen eigenen Ordner (z.B. PI1-Vorlagen), nicht in den *workspace* von Eclipse.
- Starten Sie Eclipse.
- Wählen Sie *File* → *New* → *C/C++ Project* → *C++ Managed Build*.
- Im Projektfenster geben Sie den Namen *Test_Doxygen* ein und wählen in den unteren Fenstern links *Empty Project* und rechts *MinGW GCC*, dann *Finish*.
- Klicken Sie mit der rechten Maustaste auf den Projektnamen, dann
  Import → *General* → *File System*, dann *Next*.
  Wählen Sie im folgenden Fenster bei *From directory* die Schaltfläche *Browse* und gehen Sie dann zu Ihrem Vorlagenordner. Wählen Sie aus dem Vorlagenordner
  ...\versuch00\Testprogramm_Eclipse_Doxygen, dann *OK*.
- Setzen Sie im linken Fenster ein Häkchen bei *Testprogramm_Eclipse_Doxygen*. Dadurch werden rechts alle Dateien ausgewählt, dann *Finish*. Das Projekt ist erstellt.
- Nun können Sie wie im vorigen Kapitel das Projekt ausführen.
- Fügen Sie dem Projekt über *File* → *New* → *Other* → *Other* → *Doxyfile* ein neues Doxyfile hinzu. Die Option ist nur verfügbar, wenn Sie das Eclox-Plugin installiert haben.

- Öffnen Sie die Datei und wählen Sie die Optionen wie in Abbildung 2.1 und speichern Sie die Datei. Falls der Bereich *Diagrams to generate* ausgegraut sein sollte, ignorieren Sie diese Einstellung einfach.
- Die Dokumentation wird über das blaue @ in der Menüleiste erstellt.

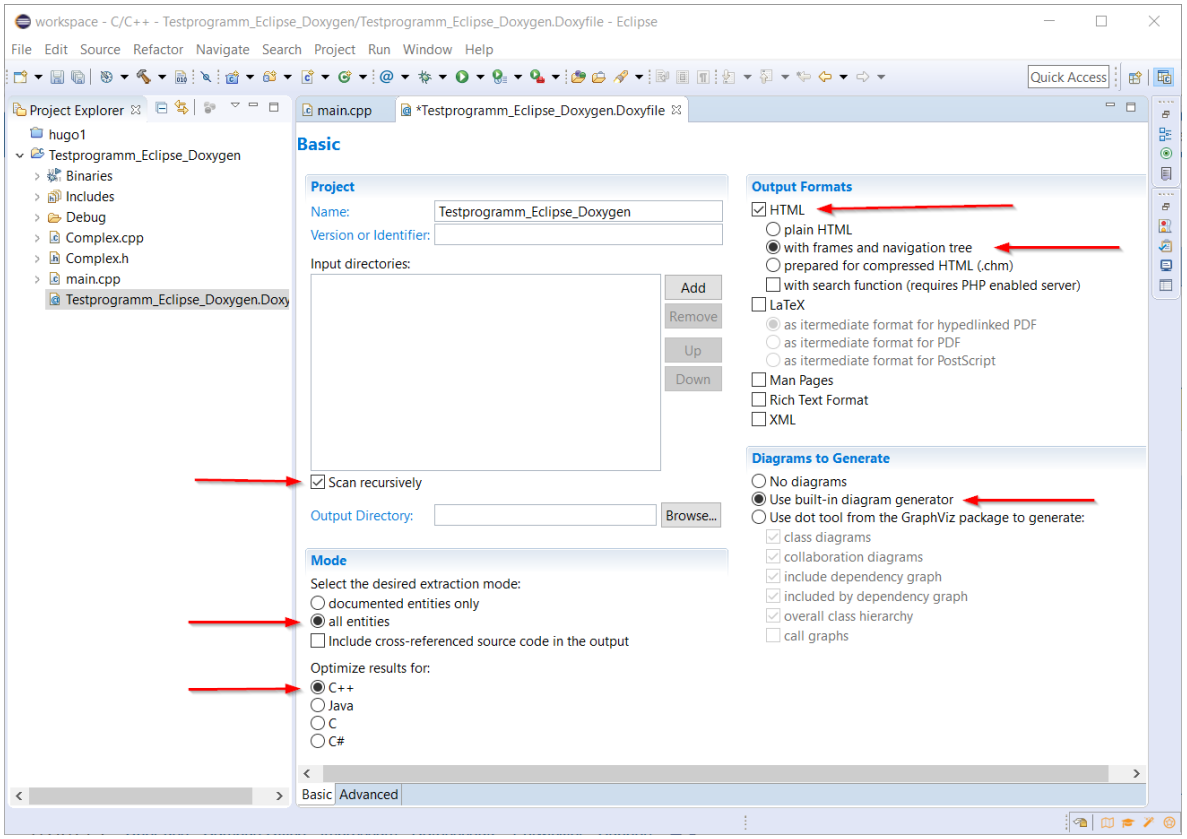

Abbildung 2.1: Doxyfile-Button und Konfiguration

Jetzt sollte ein zusätzlicher Ordner *html* in Ihrem Projekt erschienen sein. Wenn nicht, klicken Sie mit der rechten Maustaste auf Ihren Projektname und wählen Sie *Refresh*. Öffnen Sie die Datei `index.html` im Ordner `html`. Nun muss die Abbildung 2.2 erscheinen.

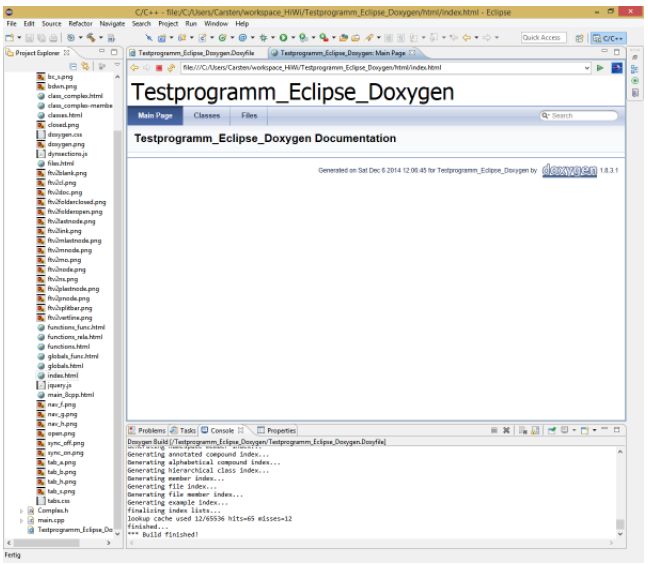

Abbildung 2.2: Testprogramm_Eclipse_Doxygen

Eine Einführung in die Kommentare finden Sie im Skript zum Praktikum.

2.3 Test Git [40:55]

Nutzen Sie für diesen Test ein Hello-World-Projekt oder das Projekt zum Test von Doxygen.

1. Öffnen Sie in einem geöffnten Projekt ein Terminal (mittels des Shortcuts STRG+Alt+T) und schreiben dort git init und bestätigen mit Enter.
2. Machen Sie einen Rechtsklick auf das zugehörige Projekt und wählen Sie *Team* → *Share Project*. Klicken Sie auf *Finish*. Neben Ihrem Projektnamen sollte ein neuer Badge aufgetaucht sein und wenn Sie auf *Team* auswählen, eine Reihe neuer Menüoptionen hinzugekommen sein.

2.4 Test Qt-Creator [46:13]

Auch für diesen Test finden Sie eine Vorlage in Ihrem Vorlagenordner.
Der Qt-Creator legt Projektdateien standardmäßig im Ordner *Dokumente* ab, und nicht in einem eigenen *workspace*. Um dies zu ändern, legen Sie zunächst einen neuen Ordner an, z.B *workspace-Qt* in Ihrem Homeverzeichnis. Im Cip-Pool muss dieser auf dem Laufwerk U: liegen.

1. Öffnen Sie das Projekt über „*Datei* → *Datei oder Projekt öffnen...*“.
2. Navigieren Sie in das Verzeichnis des Testprogramms, wählen Sie die Qt-Projektdatei *Testprogramm_ Qt.pro* aus und klicken Sie anschließend auf *Öffnen*.
3. Bestätigen Sie das Fenster mit *Projekt konfigurieren*.
4. Drücken Sie nun links unten auf den grünen Pfeil, um das Programm auszuführen. Der Qt-Creator wird es automatisch erstellen.

Ein Button mit der Aufschrift „Schließe mich“ sollte in einem Fenster erscheinen.

Falls Sie von allen Tests die erwartete Reaktion erhalten haben, sind Ihre Entwicklungsumgebungen richtig und vollständig eingerichtet.

3 Einrichten der Arbeitsumgebung mit Eclipse [17:55]

Die folgenden Einstellungen und Beschreibungen können Ihnen das Arbeiten mit Eclipse deutlich erleichern und helfen, Fehler oder unnötigen Aufwand zu vermeiden. Nehmen Sie sich im eigenen Interesse die Zeit, ihre Arbeitsumgebung einzurichten.

3.1 Codevorlagen

Im CIP-Pool finden Sie die Codevorlagen auf dem Laufwerk *P:* unter

```
P:\UserGrp\PI1\Vorlagen
```

Ansonsten laden Sie die Zip-Datei aus Moodle herunter und entpacken Sie sie in ein eigenes Verzeichnis, nicht in den Workspace von Ecplise. Für jeden Versuch gibt es einen eigenen Unterordner mit den Vorlagen.
Vergessen Sie nicht, beim Importieren die Option *Copy files* auswählen, das ist wichtig.

3.2 Startverhalten (*Launching*) des Programms konfigurieren [18:20]

Diese Einstellung kann Ihnen viel Kopfzerbrechen ersparen. Wenn Ihre Anwendung noch läuft, z.B weil sie auf eine Eingabe wartet, und Sie editieren in der Zwischenzeit Ihr Programm, weil Ihnen ein Fehler aufgefallen ist, dann kann Eclipse keine neue, ausführbare Version erstellen, da die alte Version noch ausgeführt wird.

Unter Windows ist es nicht möglich, geöffnete, ausführbare Dateien zu löschen. Der Compiler erstellt zwar eine neue Version Ihres Programms, aber der Linker, der ja die ausführbare Version erstellt, kann die Alte nicht löschen, und damit keine neue Version erstellen. In der Konsole sehen Sie dann auch eine entsprechende Fehlermeldung und es öffnet sich ein Fenster, das auf den Fehler hinweist und fragt, ob Sie trotzdem starten (*Launch*) wollen. Wenn Sie jetzt auf *Launch* klicken, wird die vorhandene, also die alte Version wieder gestartet, und Ihr Programm läuft dann zweimal. Und Sie wundern sich, warum Ihre Änderungen nicht da sind.

Das können Sie beliebig oft fortsetzen, und Ihre Änderungen erscheinen trotzdem nicht. Um dies zu vermeiden, nehmen Sie folgende Einstellung vor:
$Window \rightarrow Preferences \rightarrow Run/Debug \rightarrow Launching$
Wählen Sie unten *Launch the associated project* und setzen Sie darunter das Häckchen bei *Teminate and Relaunch while launching(...)*. Stellen Sie sicher, dass etwas weiter oben unter *General Options* der Punkt *Build (if required) before launching* ausgewählt ist.

Jetzt wird Eclipse zuerst Ihr Programm beenden, bevor dies neu kompiliert und dann ausgeführt wird.

In dem Unterordner *Launch Bar* entfernen Sie alle Häkchen, dann verschwindet in Eclipse die *Launch Bar*, die durch die zusätzlich dargestellten Build- und Run-Symbole für mehr Verwirrung als Hilfe sorgen kann.

3.3 Automatisches Speichern [20:57]

Es ist sehr nützlich, Eclipse so einzustellen, dass alle Dateien automatisch gespeichert werden, bevor eine ausführbare Datei erstellt wird. Dies verhindert eine langwierige Fehlersuche, verursacht durch eine nicht gespeicherte Datei.

Gehen Sie hierzu in die Eclipse-Einstellungen:
$Window \rightarrow Preferences \rightarrow General \rightarrow Workspace \rightarrow Build$
Aktivieren Sie die automatische Speicherfunktion: *Save automatically before build.*

Hier können Sie auch zusätzlich das Interval festlegen, in dem Ecplise automatisch geöffnete Dateien im Hintergrund sichert. Geben Sie die Anzahl Minuten unter *Workspace save interval* an.

3.4 MinGW als Default-Compiler [23:05]

Stellen Sie den MinGW GCC als Default-Compiler ein.

Gehen Sie hierzu in die Eclipse-Einstellungen:
$Window \rightarrow Preferences \rightarrow C/C++ \rightarrow New\ C/C++\ Projekt\ Wizard$
Aktivieren Sie im Tab *Preferred Toolchains* die Punkte *Empty Project* und *MinGW GCC*, dann *Make Toolchains preferred*, sodass ein graues Dreieck neben MinGW GCC erscheint. Bestätigen Sie abschließend mit *Apply* und *ok*.

3.5 Importieren von Codevorlagen

Im Laufe des Praktikums bekommen Sie oft Codevorlagen zur Verfügung gestellt, die Sie dann in ein leeres Projekt importieren und um eigenen Code ergänzen sollen.

So erstellen Sie ein leeres Projekt:

- $File \rightarrow New \rightarrow C++\ Project \rightarrow C++\ Managed\ Build \rightarrow Next$
- Tragen Sie den Projektnamen ein und wählen Sie $Empty\ Project$ und $MinGW\ GCC$, dann ok.

Nun können Sie die Vorlagen importieren.

- Im Windows-Explorer in das Vorlagenverzeichnis wechseln, zu importierende Dateien markieren und in Eclipse auf den Projektnamen ziehen.
- Im sich öffnenden Fenster die Option $Copy\ files$ auswählen, dann OK (wichtig).

Die Codedateien werden dann automatisch in das Projektverzeichnis kopiert.

3.6 Code automatisch formatieren [23:55]

Eclipse bietet dem Programmier die Möglichkeit, den Code nach verschiedenen Vorgaben (*Code-Convention*) automatisch zu formatieren. Dazu müssen Sie zuerst die richtige Vorlage auswählen. Die für das Praktikum geltende Code-Convention werden Sie noch im Kapitel drei kennenlernen, Eclipse verfügt über die entsprechende Vorgabe. Gehen Sie wie folgt vor:

- $Window \rightarrow Preferences \rightarrow C/C++ \rightarrow Code\ Style \rightarrow Formatter$

- Wählen Sie unter *Active Profile* die Option *BSD/Allman [build-in]* aus. Im Fenster darunter sehen Sie ein Beispiel, wie Ihr Code formatiert wird.
- Dann wieder *Apply* und *OK*.

Mit *Strg-Shift-F* öffnet sich ein Fenster, in dem Sie die ganze Datei oder den markierten Text auswählen können.

3.7 Perspektive wiederherstellen [24:33]

Wenn Ihnen die Perspektive durcheinander geraten ist, Fenster versehentlich verschwunden sind, können Sie die ursprüngliche Perspektive wie folgt wiederherstellen:

$Window \rightarrow Perspective \rightarrow ResetPerspective$

Alternativ können Sie auch über $Window \rightarrow ShowView$ die versehentlich geschlossen Fenster wieder öffnen. Vergessen Sie auch nicht, dass Sie oben rechts zwischen Debug- und IDE-Ansicht wechseln können, was durch ein Käfer- bzw. C-Symbol dargestellt ist.

3.8 Aktive Zeile im Editor farbig unterlegen und Tabulatorgröße setzen [22:02]

Damit Sie schneller sehen können, an welcher Stelle sich Ihr Cursor im Moment befindet, können Sie die aktive Zeile hervorheben.

- $Window \rightarrow Preferences \rightarrow General \rightarrow Editors \rightarrow Text\ Editors$
- Wählen Sie im unteren Fenster *Current line highlight* und rechts daneben die Farbe.
- Tragen Sie im Feld *Displayed tab width* "4" ein.
- Wählen Sie zusätzlich den Punkt $Insert\ spaces\ for\ tabs$ aus.
- Dann wieder *Apply* und *OK*.

3.9 Unterstützung für Doxygen

Eclipse kann Sie bei der Erstellung der Dokumentation mit *Doxygen* unterstützen. Das wird ab Kapitel 3 interessant. Wählen Sie bei

$Window \rightarrow Preferences \rightarrow C{+}{+} \rightarrow Editors$

unten rechts für *Workspace default* die Einstellung *Doxygen*. Wie Sie diese Einstellung nutzen, erfahren Sie in Kapitel 3.

3.10 Verschiedenes [25:18]

Wenn Sie einen ganzen Block auskommentieren wollen, um z.B. zu testen, ob ein Fehler in diesem Block ist, geht das recht einfach. Block markieren, dann *Strg-Shift-7*. Wenn Sie alles wieder einkommentieren wollen, genau so. Und genau so können Sie ganze Blöcke einrücken und wieder ausrückt, indem Sie *Tab* bzw. *Shift-Tab* drücken.

Eine Liste aller Tastaturbefehle bekommen Sie mit *Strg-Shift-L* angezeigt.

Mit dem Befehl *Strg-3* können Sie Befehle und Menüs auch per Suche öffnen.

4 Installation der Entwicklungsumgebungen für MacOS

Diese Installationsanleitung orientiert sich an der vom Praktikum Informatik 2 und nutzt dieselben Installationsskripte. Die Installation erfolgt über ein Terminal. Weitere Informationen zur Nutzung erfahren Sie im Anhang: Nutzung des Terminals in MacOS.

Hinweis 1: Beachten Sie, dass bei den Installationsschritten **IHRUSERNAME** mit Ihrem Benutzernamen auf Ihrem Mac ersetzt werden muss.

Hinweis 2: Bei Copy-Paste der Befehle aus dem PDF kann es je nach Reader zum Mitkopieren unerwünschter Zeichen kommen. Um diesen Fehler auszuschließen, wird das Abtippen der entsprechenden Befehlszeilen empfohlen.

4.1 Installation Xcode Command Line Tools

Öffnen Sie ein Terminal und führen Sie den folgenden Befehl wie in Abbildung 4.1 aus:

```
sudo xcode-select  --install
```

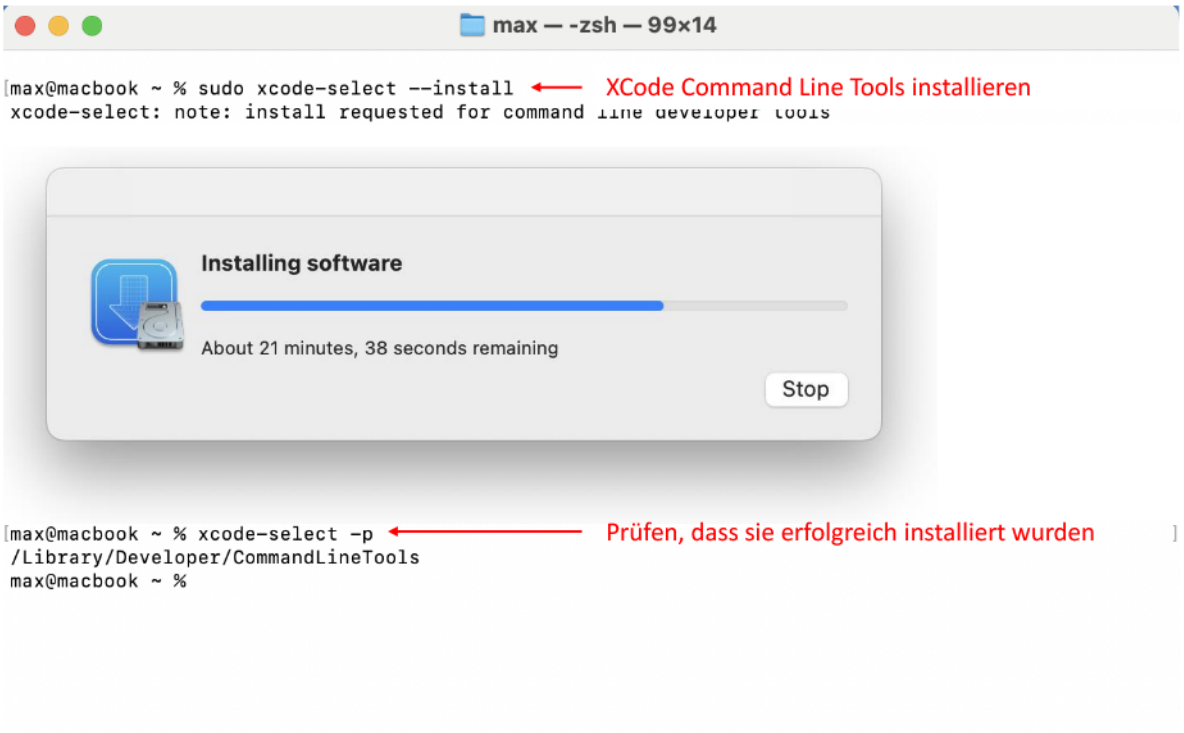

Abbildung 4.1: Installation der Xcode Command Line Tools im Terminal

4.2 Installation der Entwicklungsumgebungen für MacOS

Um Eclipse und den Debugger auf Ihrem Mac zu verwenden, benötigt es ein paar Voraussetzungen. Dazu wird Ihnen im Moodle bzw. über den Sciebo-Link eine Skript-Datei mit dem Namen *lldb-mi.sh* zur Verfügung gestellt.

1. Kopieren Sie diese Datei uf Ihrem Mac in den User-Ordner */Users/*IHRUSERNAME*/*
2. Öffnen Sie nun ein Termimal und machen die Datei mit dem Befehl

   ```
   sudo chmod +x /Users/*IHRUSERNAME*/lldb-mi.sh
   ```

   ausführbar. Das erfolgreiche Ausführen des Befehls gibt keine Antwort ans Terminal zurück. Sie können aber überprüfen, indem Sie

   ```
   ls -l /Users/*IHRUSERNAME*/lldb-mi.sh
   ```

   eingeben und dort schauen, ob die Berechtigung ausführbar mit dem Buchstaben "x" für *executable* ergänzt wurde. Danach können Sie die Datei mit

   ```
   /Users/*IHRUSERNAME*/lldb-mi.sh
   ```

   ausführen.
3. Folgen Sie den Anweisungen im Terminal. Die Batch-Datei installiert und kompiliert unter anderem den benötigten Debugger in das Verzeichnis */Users/*IHRUSERNAME*/lldbmi/*. Bei der Installation kann es dazu kommen, dass Sie mehrfach nach Ihrem Passwort gefragt werden.

Die letzten beiden Schritte sind in Abbildung 4.2 dargestellt.

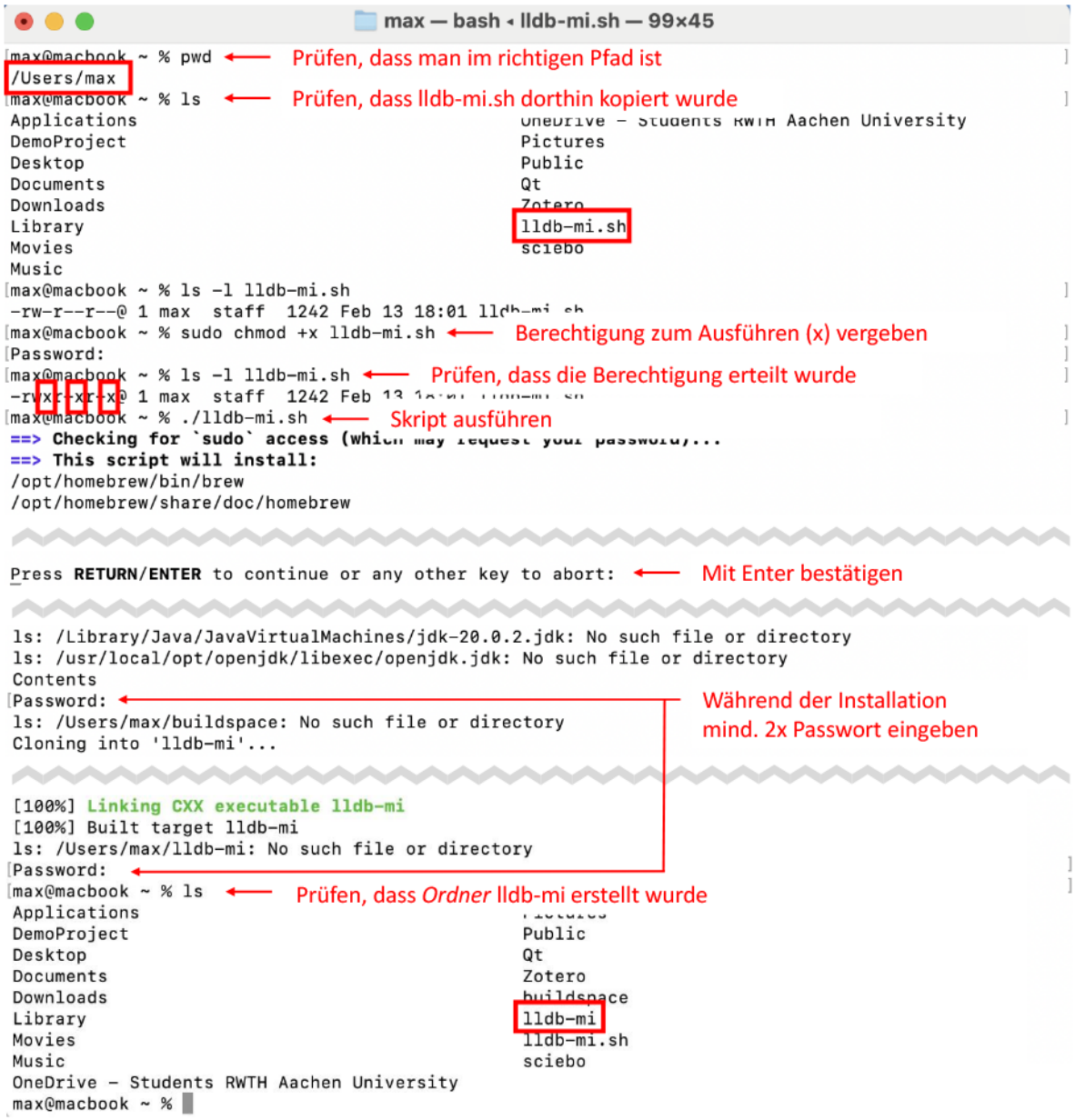

Abbildung 4.2: Installation des Debuggers lldb-mi. Anstatt der absoluten Pfade werden hier relative Pfade genutzt, was funktioniert, wenn man sich im Homeverzeichnis befindet.

4.3 Installation und Konfiguration von Eclipse

1. Laden Sie den Eclipse Installer für Ihrem Mac herunter und führen Sie das Installationspaket aus:
   a) für den Mac mit **m*-Chip** laden Sie bitte folgenden Installer herunter:`https://www.eclipse.org/downloads/download.php?file=/oomph/epp/2024-09/R/eclipse-inst-jre-mac-aarch64.dmg`
   b) für alle anderen Macs (mit Intel-Chip) nutzen Sie folgenden Installer: `https://www.eclipse.org/downloads/download.php?file=/oomph/epp/2024-09/R/eclipse-inst-jre-mac64.dmg`
2. Starten Sie den Eclipse Installer aus dem Paket und wählen Sie *Eclipse IDE for C/C++ Developers* aus.
3. Wählen Sie in der nächsten Abfrage einen geeigneten Installationspfad (der Pfad kann unverändert bleiben) aus und bestätigen Sie die Installation (Auswahl der Java 17+ VM bleibt unverändert).
4. Nach erfolgreicher Installation können Sie den Installer mit *Launch* beenden und Eclipse starten.
5. Beim Start von Eclipse werden Sie nach dem Ort für den Workspace gefragt. In dem Workspace speichert Eclipse alle Projekte und die dazugehörigen Metadaten ab. Wählen Sie bitte einen entsprechenden Pfad auf Ihrem System aus und bestätigen Sie die Auswahl mit *Launch*.
6. Testen Sie nun die Installation von Eclipse, indem Sie ein neues Projekt in Eclipse anlegen. Dazu wählen Sie *File* → *New* → *Project* → *C/C++ Project* → *C++ Managed Build* aus und klicken Sie auf *Next*. Geben Sie nun einen Projektnamen (z.B. *HelloWorld*) ein, wählen Sie als Project type *Hello World C++ Project* und als Toolchain *MacOSX GCC* aus. Die korrekte Auswahl ist in Abbildung 4.3 abgebildet. Legen Sie das Projekt mit *Finish* an. Nun haben Sie Ihr erstes Projekt mit einem kleinen Beispielprogramm angelegt. Damit Sie dieses kleine Test-Programm ausführen können, müssen Sie es zuerst kompilieren. Dazu wählen Sie *Project* → *Build Project* aus. Nach dem der Build-Prozess abgeschlossen ist, können Sie das Projekt über *Run* → *Run As* → *2 Local C/C++ Application* ausführen. Nun erscheint in der Konsolen-Ausgabe *!!!Hello Word!!!*.
7. Um evtl. falsche Fehlermeldungen von Eclipse auszublenden, öffnen Sie das Menü *Project* → *Properties* → *C/C++ General* → *Code Analysis* → *Use project settings* und unchecken Sie *Syntax and Semantic Errors*. Beachten Sie, dass Sie diese Einstellungen für jedes (Teil-)-Projekt in Eclipse wiederholen müssen.

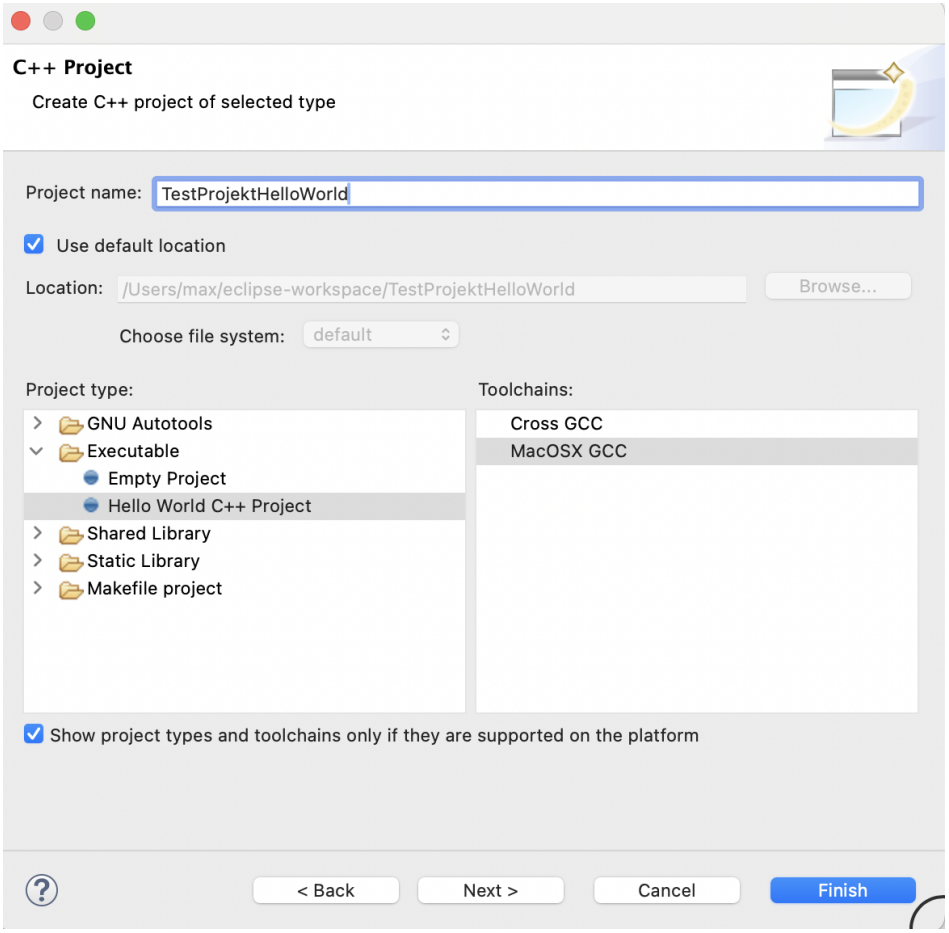

Abbildung 4.3: Erstellen eines neuen C++-Projekts in macOS. Anstatt der für Windows spezifischen Toolchain MinGW wird die für macOS spezifische Toolchain MacOSX GCC ausgewählt.

4.4 Debugger Konfiguration

1. Öffnen Sie Eclipse und öffnen Sie über das Kontextmenü *Help* → *Install New Software* den Paket-Manager von Eclipse. Hier wählen Sie bei *Work with CDT...* aus. Markieren Sie in der Auswahl unter *CDT Optional Features* das Paket *C/C++ LLDB Debugger Integration (experimental)* aus und bestätigen die Installation mit *Next*, *Finish* und *Restart Now*. Die Auswahl ist in Abbildung 4.4 zu sehen.
2. Gehen Sie nun in die Einstellungen unter *Eclipse* → *Settings*. Wählen Sie dort unter *Run/Debug* → *Launching* → *Default Launchers* aus. Klicken Sie unter *C/C++ Application* auf *Debug* und wählen dort unter *Preferred Launcher* die Option *LLDB-MI Debug Process Launcher* aus. Das Ergebnis ist in Abbildung 4.5 zu sehen. Bestätigen Sie anschließend mit *Apply and Close*.
3. Gehen Sie wieder in die Einstellungen unter *Eclipse* → *Settings*. Wählen Sie dort unter *C/C++* → *Debug* → *LLDB*. Unter *LLDB comand* klicken Sie auf den Button *Browse*, sodass dort der Pfad */Users/*IHRUSERNAME*/lldb-mi/lldb-mi* steht. Das Ergebnis ist in Abbildung 4.6 zu sehen. Bestätigen Sie mit *Apply and Close*, dann können Sie mit dem Debugging beginnen. Beim Starten des Debuggers kann es zu der Abfrage von Benutzername und Passwort Ihres Macs kommen.

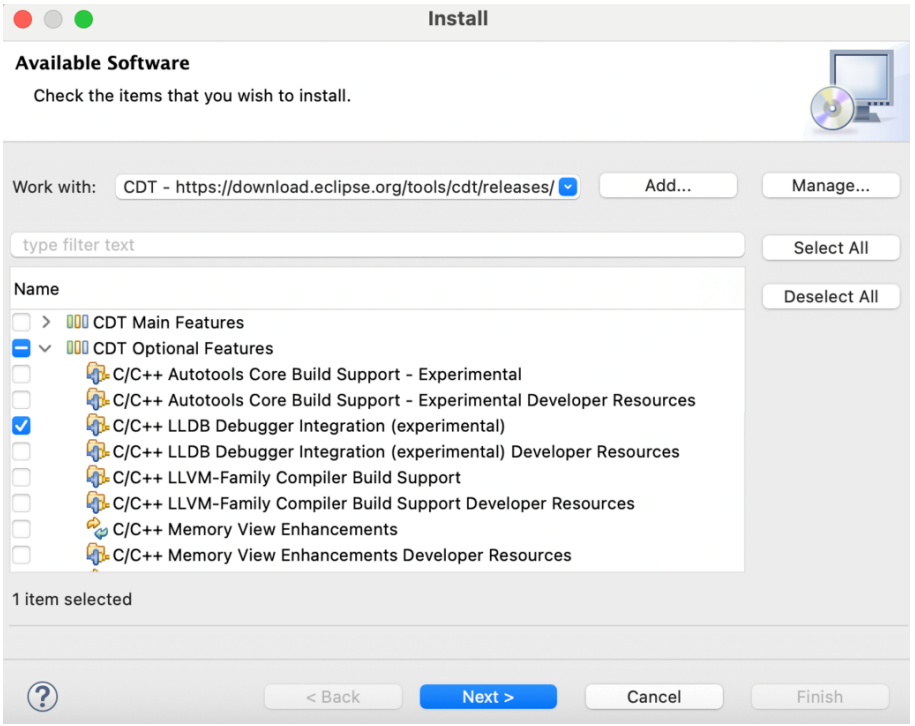

Abbildung 4.4: Installation der LLDB Debuger Integration

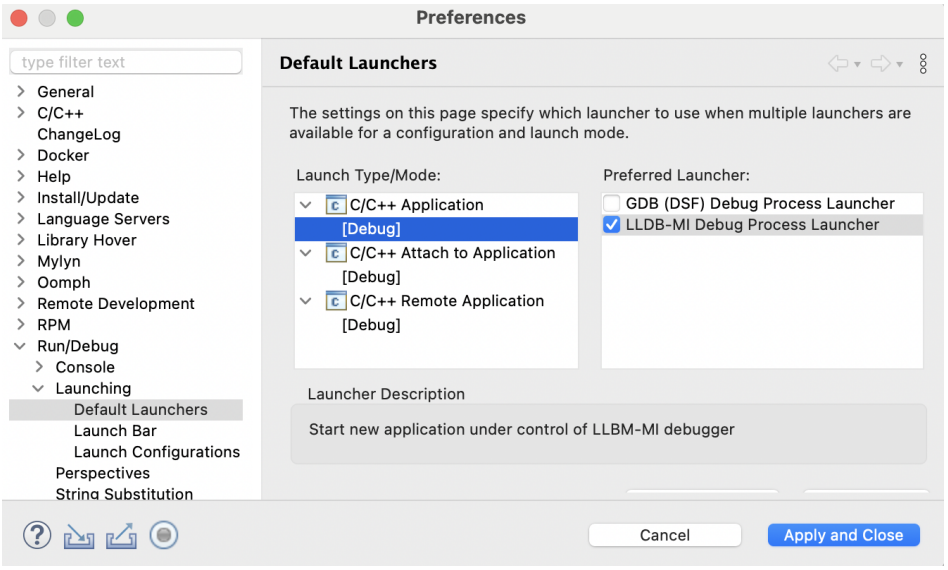

Abbildung 4.5: Einstellung in Eclipse, damit der Debugger Launcher LLDB-MI als Default anstatt GDB ausgewählt wird.

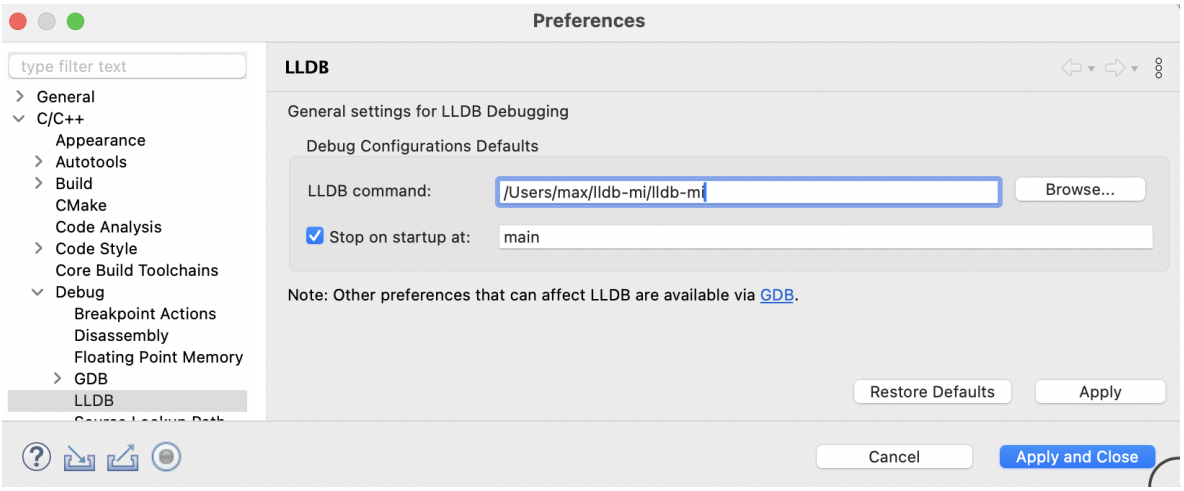

Abbildung 4.6: Einstellung in Eclipse, damit der Debugger korrekt ausgewählt wird.

4.5 Doxygen

Es gibt verschiedene Wege Doxygen zu installieren: über das Disk Image File von der offiziellen Doxygen-Website[1],während der Installation von Eclox oder über den Packetmanager *Homebrew*. Diese Anleitung beschreibt letzteres.

1. Öffnen Sie ein Terminal und installieren Sie Doxygen mit dem folgenden Command:

   ```
   brew install --cask doxygen
   ```

   Der Packetmanager *brew* wurde in Abschnitt 4.2 installiert. Falls es eine Fehlermeldung gibt, dass *brew* nicht gefunden werden kann, fügen Sie den Packetmanager zu Ihrem Pfad hinzu wie in Abbildung 4.7 gezeigt:

   ```
   echo 'eval "$(/opt/homebrew/bin/brew shellenv)"' >> ~/.zprofile
   eval "$(/opt/homebrew/bin/brew shellenv)"
   ```

2. Starten Sie Doxygen beispielsweise mit dem folgenden Command:

   ```
   open /Applications/Doxygen.app
   ```

3. Falls Ihr Computer eine Warnung ausgibt, gehen Sie zu *Settings* → *Privacy and Security* und scrollen Sie zu *Doxygen ...* wie in Abbildung 4.8 gezeigt. Sie können dort mit einem Klick auf *Open Anyway* die Sicherheitseinstellung Ihres Computers überschreiben.

4. Installieren Sie das Doxygen Plugin für Eclipse wie in Abschnitt 1.4.1 beschrieben. Starten Sie Eclipse anschließend neu.

5. Fügen Sie den Pfad von Doxygen in Eclipse, damit dieses von Eclox gefunden werden kann. Gehen Sie dabei wie in Abbildung 4.9 gezeigt zu *Einstellungen* → *Doxygen* und klicken Sie auf *Add..* und wählen dort den Pfad

   ```
   /Applications/Doxygen.app/Contents/Resources
   ```

   aus. Anschließend klicken Sie auf den Haken, sodass der benutzerdefinierte Speicherort genutzt wird und bestätigen mit *Apply and Close*.

6. Testen Sie Ihre Doxygen-Installation wie in Abschnitt 2.2 beschrieben.

[1] `https://www.doxygen.nl/files/Doxygen-1.13.2.dmg`

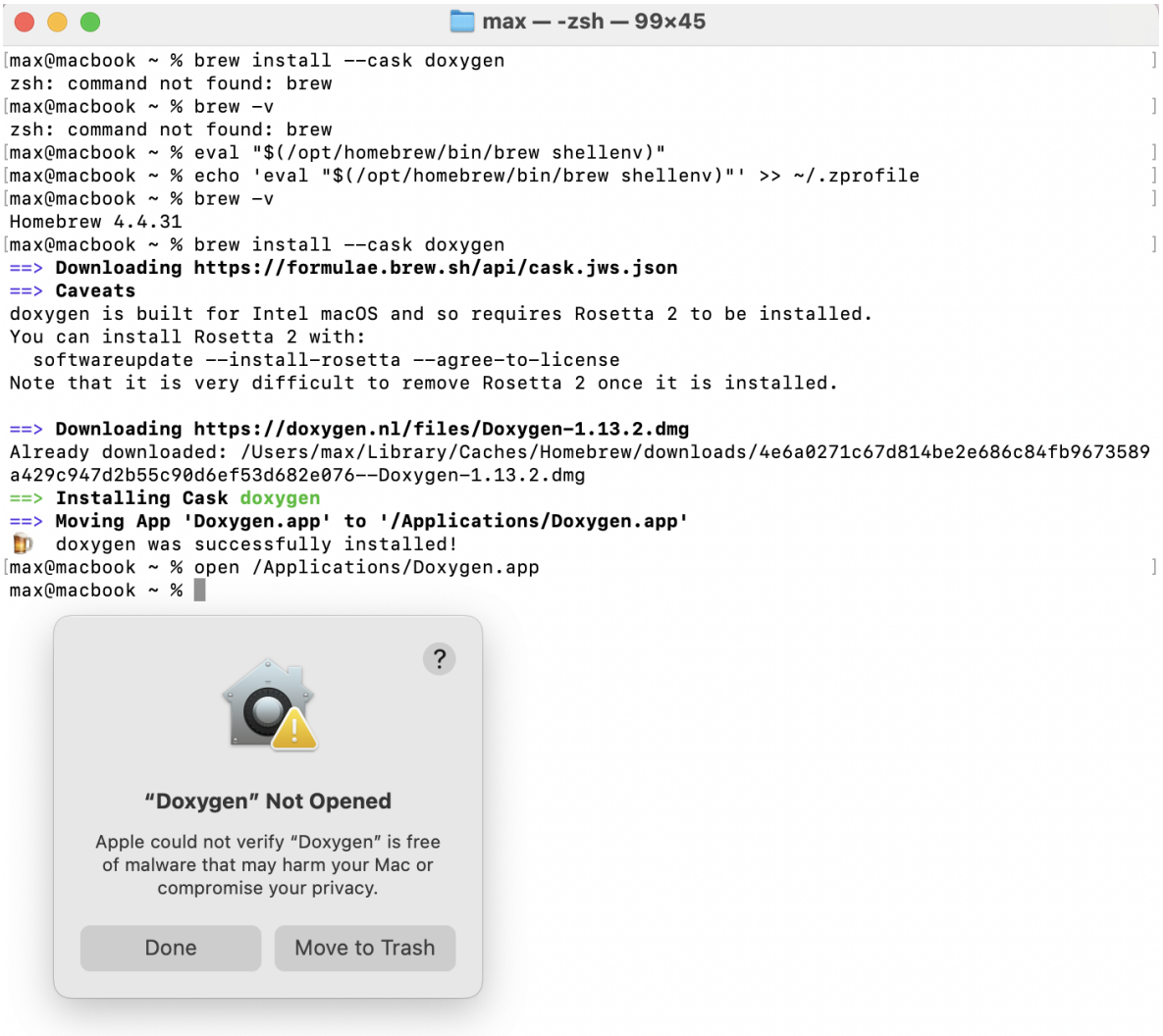

Abbildung 4.7

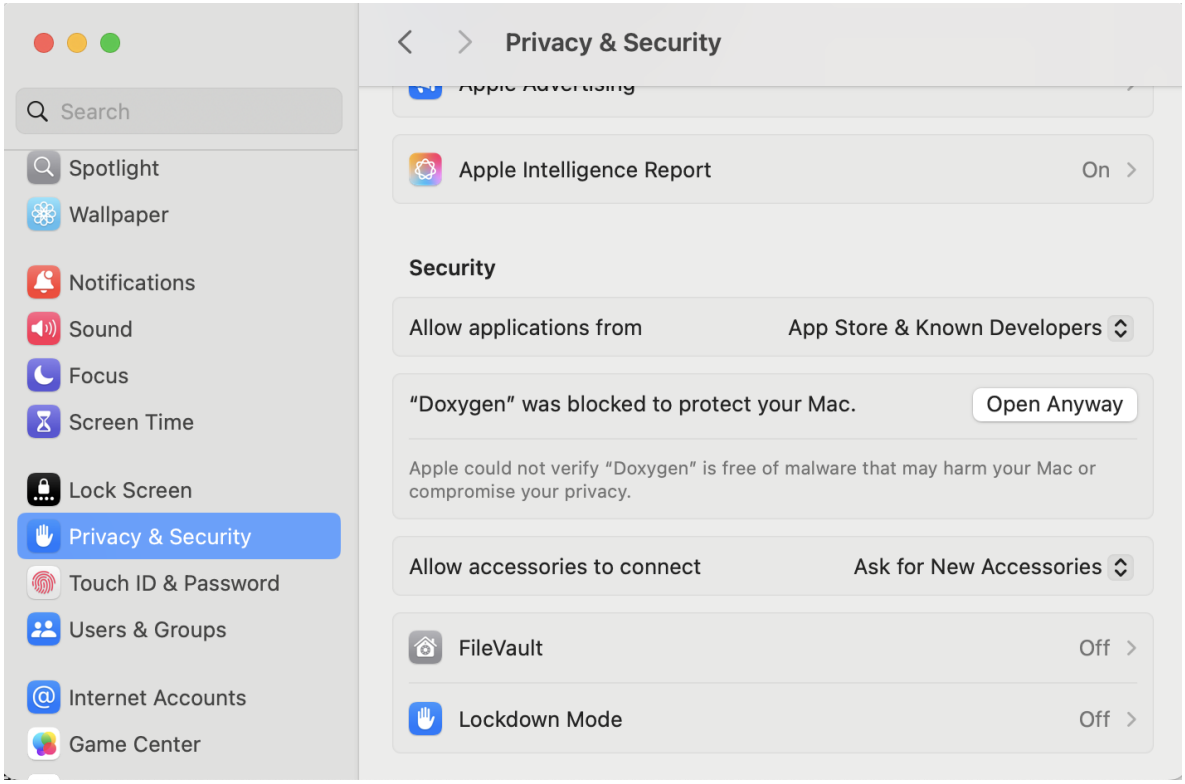

Abbildung 4.8

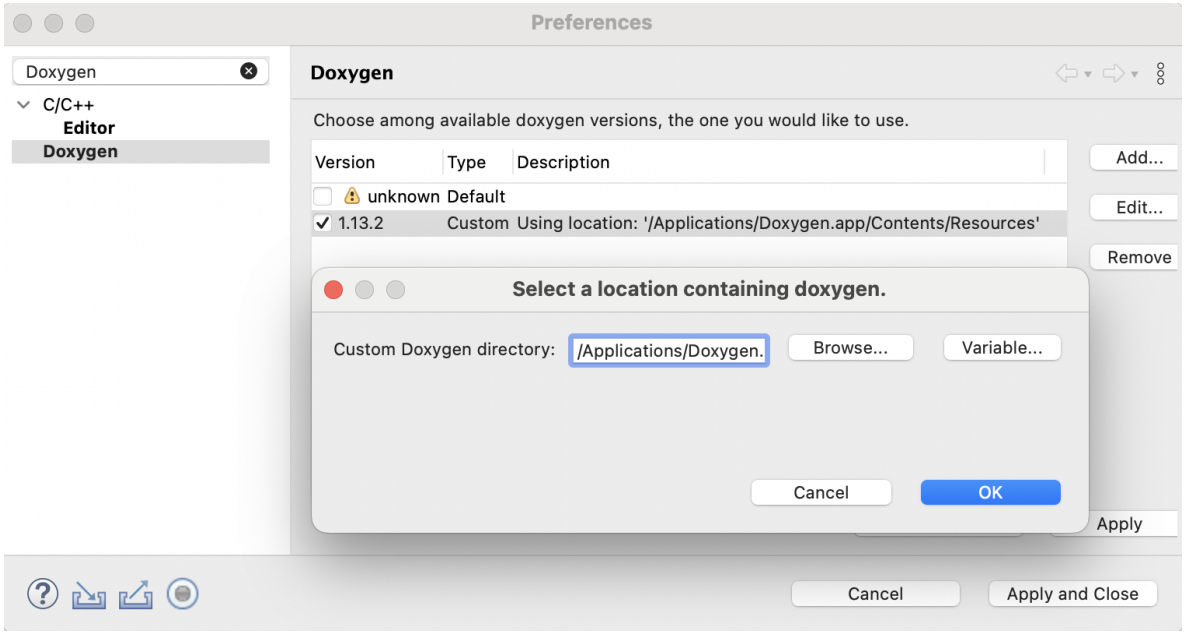

Abbildung 4.9

4.6 Qt-Creator

Öffnen Sie den Qt-Installer und klicken Sie sich durch

1. Auf der Seite *Anmelden*, klicken Sie auf *Registrieren* und erstellen Sie einen Qt-Account.
2. Lesen Sie die Bedingungen zu Open Source-Verpflichtungen und wenn Sie diesen zustimmen, aktivieren Sie die beiden Häckchen.
3. Wählen Sie in den *Installationseinstellungen* das vorausgewählte Verzeichnis */Users/*IHRUSERNAME*/Qt*, sowie die vorausgewählte Einstellung *Qt 6.x für Desktop-Entwicklung*. Wählen Sie zusätzlich die Option *Benuterdefinierte Option*.
4. Auf der Seite *Anpassen* wählen Sie unter *Qt → Qt 6.x.y* die Option *Additional Libraries* ab. Und unter *Qt → Build Tools* können Sie die Optionen *CMake* und *Ninja* deaktivieren. Die Auswahl für die Qt-Version ist auch nochmal in Abbildung4.10 zu sehen.
5. Wenn Sie mit den *Lizenzabkommen* einverstanden sind bestätigen Sie diese.
6. Behalten Sie den Startmenüordner bei und klicken Sie abschließend auf *Installieren*. Die Gesamtgröße sollte etwa 2.7 GB betragen.
7. Testen Sie abschließend die erfolgreiche Installation wie in Abschnitt 2.4 beschrieben.

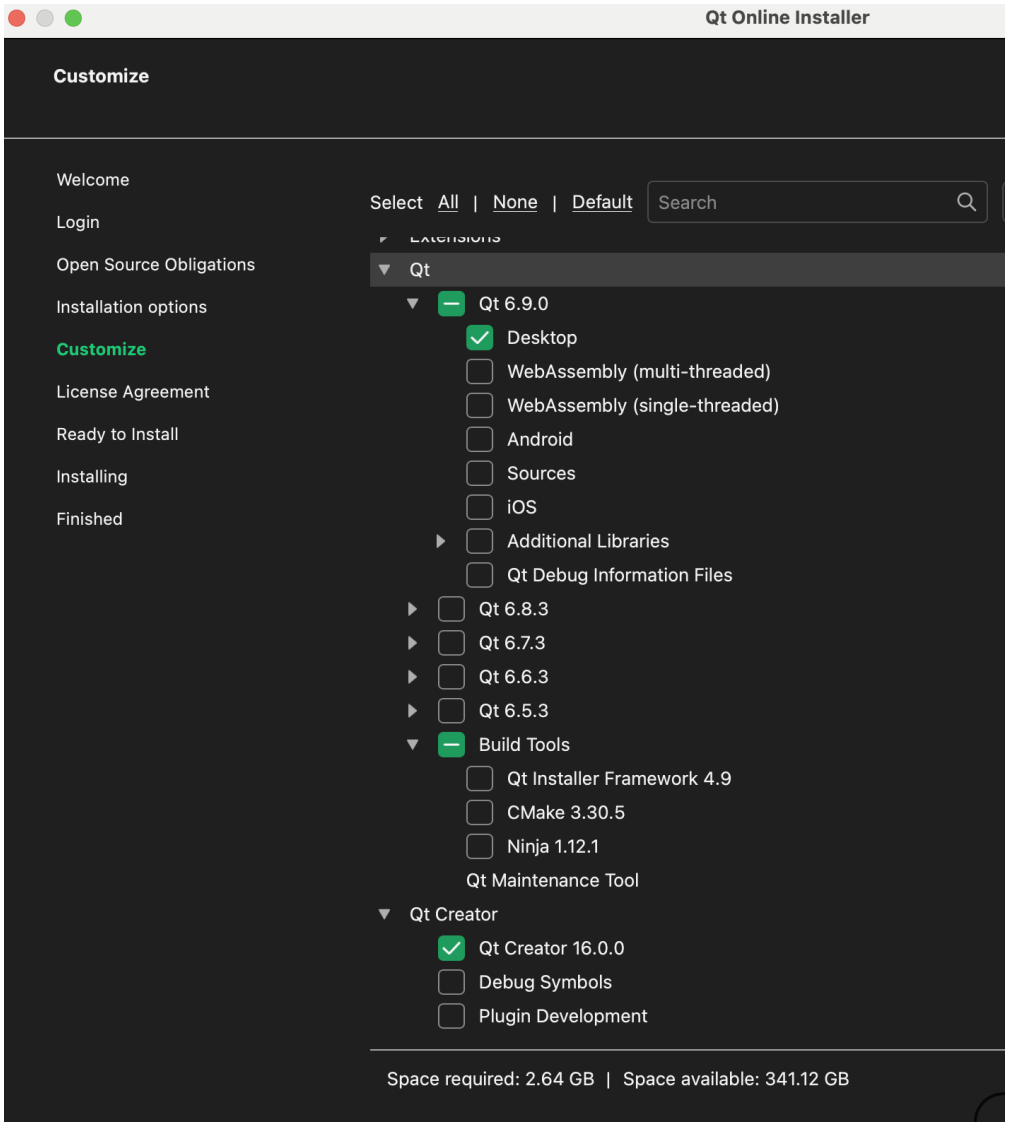

Abbildung 4.10: Auszuwählende Komponenten im Qt Installer. Die Versionsnummern könnten bei neueren Versionen anders aussehen

4.7 Software deinstallieren

Wenn Sie die Software deinstallieren möchten, ist hier eine Übersicht der Befehle, welche Sie im Terminal eingeben können. Bitte beachten Sie, dass `~/` stellvertretend für `/Users/*IHRUSERNAME*/` genutzt werden kann.

- Wenn Sie die XCode Command Line Tools deinstallieren möchten, führen Sie folgende Befehle aus:

  ```
  sudo rm -rf /Library/Developer/CommandLineTools
  sudo xcode-select --reset
  ```

  Bitte beachten Sie auch, dass Eclipse ohne *clang* aus den XCode Command Line Tools keine Programme mehr kompilieren kann.

- Um Eclipse zu deinstallieren, entfernen Sie den folgenden Ordner:

  ```
  sudo rm -rf /Applications/Eclipse.app
  sudo rm -rf ~/eclipse
  ```

- Für Doxygen, Java, Cmake und LLVM wurde der Packetmanager Homebrew verwendet. Entsprechend können Sie diese wie folgt deinstallieren und entsprechende Verzeichnisse löschen:

  ```
  brew uninstall --cask doxygen
  brew uninstall java
  sudo rm -rf /Library/Java/JavaVirtualMachines/openjdk.jdk
  brew uninstall cmake
  brew uninstall llvm
  ```

  Bitte beachten Sie auch, dass Eclipse ohne Java nicht mehr funktioniert.

- Wenn Sie Homebrew selber löschen möchten, können Sie das wie folgt tun

  ```
  sudo rm -rf /opt/homebrew /usr/local/Homebrew ~/.brew
  ```

- Der Debugger lldb-mi wurde mittels LLVM gebaut und dafür verschiedene Verzeichnisse angelegt. Diese können wie folgt gelöscht werden:

  ```
  sudo rm -rf ~/buildspace ~/lldb-mi
  ```

- Um QT zu deinstallieren, folgen nutzen Sie am besten das Qt-Maintenance-Tool, welches Sie beispielsweise über den folgenden Befehl öffnen können:

  ```
  open ~/Qt/MaintenanceTool.app
  ```

  Es öffnet sich eine GUI ähnlich dem Installer, in welcher Sie links den Menüpunkt zum Deinstallieren sehen.

Anhang: Nutzung des Terminals in MacOS

Sie können ein Terminal öffnen, indem Sie die Suche mit *Cmd + Space* öffnen und dort *Terminal* eingeben. Mit dem Kürzel ~ wird auch das Homevereichnis bezeichnet, was abgekürzt für `/Users/*IhrUsername*` steht.

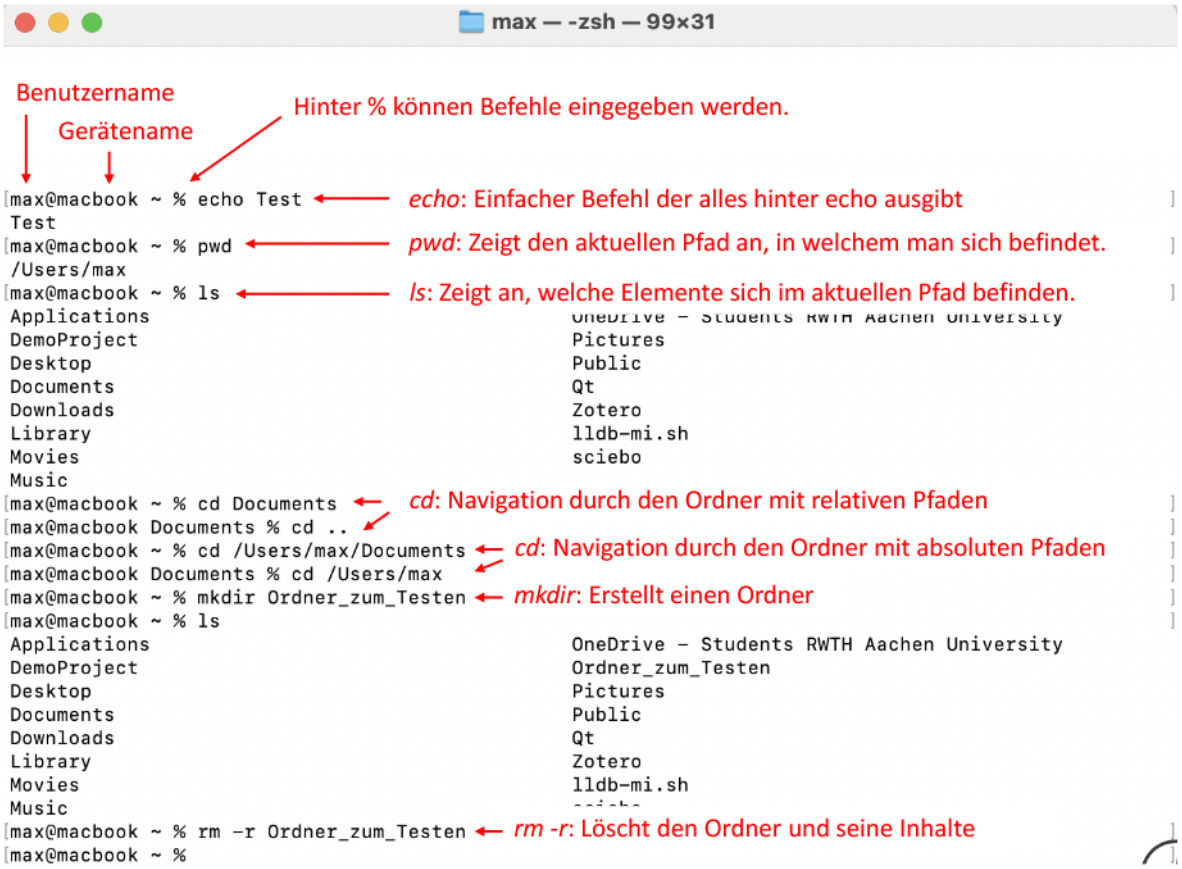

Abbildung 11: Nutzung des Terminals in MacOS

Index