

Inhaltsverzeichnis

I Einstieg in Eclipse und CDT	1
1 Installation	3
1.1 C- und C++-Compiler	3
1.2 Das richtige Eclipse-Archiv finden	4
1.3 Installation unter Linux.....	6
1.3.1 Compiler und Tools	7
1.3.2 Java	7
1.3.3 Eclipse.....	8
1.4 Installation unter Windows	8
1.4.1 Compiler und Tools	8
1.4.2 Java	13
1.4.3 Eclipse.....	13
1.5 Die Eclipse.ini-Datei	14
1.6 Willkommen.....	16
1.7 Eclipse erweitern	17
1.7.1 CDT nachträglich installieren	18
1.7.2 Update-Sites	20
1.7.3 Features verwalten	22
1.7.4 Die CDT-Features im Detail	23
1.7.5 Eclipse Marketplace	24
1.7.6 p2	27
2 Erste Schritte	29
2.1 Konzepte des Eclipse-Frameworks	29
2.1.1 Workspace	29
2.1.2 Ressourcen	30
2.1.3 Eclipse-Workbench.....	30
2.1.4 Menüleiste	30
2.1.5 Werkzeugleiste	31
2.1.6 Statusleiste	32
2.1.7 Editoren und Views.....	33
2.1.8 Perspektiven	34

2.2	Das erste Projekt	36
2.2.1	Eclipse anpassen	36
2.2.2	Vorgehensweise	38
2.2.3	Ein neues Projekt anlegen	38
2.2.4	Die Quelldatei erzeugen	38
2.2.5	Übersetzen	40
2.2.6	Ausführen	42
2.2.7	Programmfehler aufspüren	42
2.2.8	Zusammenfassung und Ausblick	44
2.3	Das Eclipse-Hilfesystem	45
2.3.1	Dokumentationsaufbau	45
2.3.2	Hilfefenster	46
2.3.3	Kontextbezogene Hilfe	48
2.4	Eclipse über die Tastatur steuern	49
2.5	Nützliche Einstellungen	50
2.5.1	Voreinstellungsdialog	50
2.5.2	Hierarchie der Voreinstellungen	51
2.5.3	Allgemeine Workspace-Einstellungen	52
2.5.4	Workspace-Auswahl beim Starten und Beenden	54
2.5.5	Netzwerkeinstellungen	55
2.5.6	Einstellungen für SSH-Client	55
2.5.7	Passwortspeicher	57
2.5.8	Komponenten (de-)aktivieren	57
2.6	Die Benutzerschnittstelle anpassen	58
2.6.1	Werkzeug- und Menüleiste anpassen	59
2.6.2	Perspektiven verwalten	62
2.6.3	Tastatursteuerung anpassen	62
2.6.4	Erscheinungsbild	64
2.7	Verzeichnislayout auf dem Datenträger	65
2.7.1	Programmverzeichnis	65
2.7.2	Workspace	67

II Entwicklung mit CDT 69

3	Projekte erstellen und bearbeiten	71
3.1	Konzepte von CDT	72
3.1.1	Parser	72
3.1.2	Indexer	72
3.1.3	Piktogramme	73
3.1.4	Kurze Einführung in das Build-System	74

3.2	C- oder C++-Projekte erzeugen	75
3.2.1	Projekte anlegen	76
3.2.2	Dateien erzeugen	79
3.2.3	Verzeichnisse hinzufügen	84
3.2.4	Vorhandene Projekte importieren	85
3.2.5	Projekt entfernen	88
3.2.6	Projekttyp ändern	88
3.3	Eigenschaften von Ressourcen	89
3.4	Projekt: Dirscanner	91
3.4.1	Boost installieren	92
3.4.2	Dirscanner entwickeln	94
3.5	Quelltexte bearbeiten	99
3.5.1	Grundsätzlicher Aufbau eines Editorfeldes	99
3.5.2	Textpassagen markieren	99
3.5.3	Quelltext formatieren	100
3.5.4	Editierhilfen	101
3.5.5	Präsentation der Quelltexte	101
3.5.6	Annotationen	105
3.5.7	Code-Analyse	107
3.5.8	Navigation	109
3.5.9	Content-Assistenz	110
3.5.10	Makros untersuchen	113
3.5.11	Suchen und Ersetzen	115
3.5.12	Include-Direktiven organisieren	118
3.5.13	Setters und Getters erzeugen	120
3.5.14	Methode implementieren	120
3.5.15	Kommentare für externe Dokumentationswerkzeuge ..	121
3.5.16	Bookmarks	123
3.5.17	Makefile	124
3.5.18	Undo/Redo-Funktion	124
3.6	Dateienvergleich	125
3.6.1	Zwei Dateien vergleichen	125
3.6.2	Drei Dateien vergleichen	128
3.6.3	Lokale Historie	128
3.7	Working-Sets	129
3.7.1	Ein Working-Set erstellen	130
3.7.2	Working-Sets einschalten	132
3.7.3	Workbench-Working-Sets	132
3.8	Tasks	133
3.8.1	Tasks hinzufügen	133
3.8.2	Tasks mit Schlüsselwörtern	134
3.8.3	Tasks anzeigen lassen	134

3.9	Refactoring	135
3.9.1	Ressourcen umbenennen	136
3.9.2	Bezeichner umbenennen	136
3.9.3	Konstanten extrahieren	140
3.9.4	Lokale Variablen extrahieren	141
3.9.5	Eine Funktion oder Methode extrahieren	142
3.9.6	Funktionsdefinition umplatzieren	144
3.9.7	Methoden verstecken	144
3.9.8	Historie der Refactorings	145
3.10	Parser- und Indexerdetails	146
3.10.1	Parser	146
3.10.2	Indexeroptionen anpassen	148
3.10.3	Weitere Indexer-Parameter	151
3.10.4	Indexdatenbank analysieren	152
3.10.5	Probleme mit dem Indexer	153
3.11	Nützliche Views der C/C++-Perspektive	154
3.11.1	Outline	154
3.11.2	Include Browser	155
3.11.3	Call Hierarchy	157
3.11.4	Type Hierarchy	158
3.12	Navigations-Views	159
3.12.1	Navigator	160
3.12.2	C/C++ Project	161
3.12.3	Project Explorer	162
3.13	Ressourcen finden	163
3.14	Elemente finden	163
3.15	Suchen und Ersetzen	165
3.15.1	File Search	166
3.15.2	C/C++ Search	167
3.16	Eine Frage des Stils	169
3.16.1	Allgemeine Einstellungen	169
3.16.2	Quelltextformatierung	169
3.16.3	Quelltextschablonen	171
3.16.4	Namens- und Bezeichnerkonventionen	172
3.16.5	Include-Direktiven organisieren	173
3.17	Editorschablonen	174
3.18	Projektspezifische Metadateien	176
3.18.1	Metadateien im Projektverzeichnis	176
3.18.2	Metadateien im Workspace-Verzeichnis	177

4	Der Build-Prozess	179
4.1	Builder	180
4.1.1	Die vier Modi eines Builders	180
4.1.2	Builder-Assoziationen	181
4.1.3	Builder-Konfiguration	182
4.1.4	Zwei Builder für C/C++-Projekte.....	183
4.2	Build-Variablen	184
4.2.1	Eigenschaften von Build-Variablen	184
4.2.2	Build-Variablen benutzen	185
4.2.3	Gültigkeitsbereich von Build-Variablen	185
4.2.4	Workspace-weite Build-Variablen definieren	187
4.2.5	Umgebungsvariablen als Build-Variablen	188
4.2.6	Dynamische Build-Variablen	188
4.3	Kontrolle über Managed Build	189
4.3.1	Konfigurationen verwalten	189
4.3.2	Quellverzeichnisse	189
4.3.3	Dateien vom Build-Prozess ausschließen	191
4.3.4	Custom Build	191
4.4	Einstellungen für C/C++-Projekte	192
4.4.1	C/C++ Build.....	193
4.4.2	Paths and Symbols	196
4.4.3	Preprocessor Include Paths, Macros	197
4.4.4	Environment	201
4.4.5	Logging	203
4.4.6	Settings	203
4.4.7	Tool Chain Editor	209
4.4.8	Build Variables	212
4.5	Einstellungen für Ressourcen	212
4.5.1	Settings	213
4.5.2	Tool Chain Editor	215
4.5.3	Path and Symbols	216
4.5.4	Preprocessor Include Paths, Macros	216
4.6	Makefile-Projekte	217
4.6.1	Der <i>Make Targets-View</i>	217
4.6.2	Make Targets hinzufügen	218
4.6.3	Make Targets aufrufen	219
4.6.4	Make Targets entfernen	219
4.6.5	Besonderheiten	220
4.7	Compiler-Ausgaben	220
4.7.1	Console	220
4.7.2	Voreinstellungen zur Build-Konsole	222
4.7.3	Wie CDT Kommandoausgaben verarbeitet	222
4.7.4	Error Parser festlegen.....	223

4.7.5	Übersetzungsfehler finden	225
4.7.6	Fehler filtern	226
4.8	Code-Generatoren verwenden	227
4.8.1	Beispiel: mathematische Ausdrücke auswerten	228
4.8.2	Bison installieren	230
4.8.3	Projekt erstellen und Quelltexte einpflegen	230
4.8.4	Einstellungen anpassen	233
4.8.5	Projekt übersetzen und ausführen	235
4.8.6	Als Projekt mit Makefile	235
4.8.7	Fazit	236
4.9	Cross-Compiling	237
4.9.1	Cross-Compiling für Make-Projekte	237
4.9.2	Cross-Compiling für Managed Build	237
4.9.3	Beispiel: AmigaOS4	238
4.9.4	Cross-Compiling-Plugin	240
4.10	Remote Compiling	240
4.10.1	Das Programm ssh einrichten	241
4.10.2	Das Dateisystem einrichten	242
4.10.3	Ein Remote-Build-Kommando erstellen	243
4.10.4	Remote Compiling in Eclipse anwenden	244
4.11	Microsoft Visual C++ einbinden	245
4.11.1	Visual C++ installieren	246
4.11.2	Das Projekt einrichten	248
4.11.3	Den Build-Prozess anpassen	249
5	Ausführen und Debugging	255
5.1	Programmstartdialoge	256
5.1.1	Main	258
5.1.2	Arguments	258
5.1.3	Environment	259
5.1.4	Common	259
5.2	Doxxygen einbinden	261
5.2.1	Konfigurationsdatei anlegen	261
5.2.2	Doxxygen-Tool einrichten	262
5.2.3	Doxxygen aufrufen	263
5.3	Die Launch-Konsole	264
5.4	Programme debuggen	265
5.4.1	Programme im Debug-Modus starten	265
5.4.2	Debug-Launcher	266
5.4.3	Debugger konfigurieren	267
5.4.4	Debug-Perspektive	269
5.4.5	Quelltexte lokalisieren	269
5.4.6	Der Debug-View	271

5.5	Den Programmablauf gezielt unterbrechen	275
5.5.1	Breakpoints-View	275
5.5.2	Zeilen-Breakpoints	277
5.5.3	Funktions-Breakpoints	278
5.5.4	Data-Breakpoints	279
5.5.5	Address-Breakpoints	280
5.5.6	Event-Breakpoints	280
5.5.7	Eigenschaften von Breakpoints	281
5.5.8	Dynamisches Printf.....	284
5.6	Nützliche Views beim Debuggen	285
5.6.1	Variables	285
5.6.2	Expressions	288
5.6.3	Register	290
5.6.4	Disassembly	291
5.6.5	Signale	292
5.6.6	Memory	293
5.6.7	Executables.....	296
5.6.8	OS Resources	297
5.7	Reverse Debugging	297
5.7.1	Reverse Debugging einschalten	298
5.7.2	Kontrollfluss steuern	298
5.7.3	Beispiel: Binäre Suche mit einem Vergleich	299
5.8	Unit Testing	302
5.8.1	Boost-Test-Library benutzen	302
5.8.2	Test starten	304
5.9	Auf entfernten Rechnern debuggen	305
5.10	Tracepoints.....	307
5.10.1	Tracepoints hinzufügen	307
5.10.2	Tracepoint-Aktionen	307
5.10.3	Traces auswerten	309
5.11	Launch Groups	310

III	Weitere nützliche Plugins	313
------------	----------------------------------	------------

6	Versionsverwaltung mit EGit	315
6.1	Grundlagen	316
6.1.1	Motivation.....	316
6.1.2	Begriffe	318
6.1.3	Konzepte von Git.....	319
6.2	EGit installieren	320
6.3	EGit konfigurieren	322
6.4	Die Perspektive Git	323

6.5	Bestehende Projekte mit Git assoziieren	324
6.6	Ein entferntes Repository klonen	325
6.7	Projekt in den Workspace importieren	327
6.8	Dekorationen	328
6.9	Mit Projekten unter Git arbeiten	329
6.10	Versionen vergleichen	337
6.11	Änderungsanzeige mit Git	338
6.12	Historie	338
6.13	Arbeiten mit Entwicklungszweigen	340
6.14	Rebase	344
6.15	Konflikte lösen	347
6.16	Stash	348
6.17	Weitere Informationen zu Git	349
7	Mylyn	351
7.1	Mylyn installieren	352
7.2	Einführendes Beispiel	353
7.3	Konzepte	356
7.4	Task Connectors	357
7.4.1	Bugzilla	357
7.4.2	SourceForge	360
7.4.3	GitHub	363
7.5	Weitere Konnektoren	365
8	Target Management	367
8.1	Target-Management-Plugin installieren	367
8.2	Konzepte	368
8.3	Die Perspektive <i>Remote System Explorer</i>	369
8.4	Der <i>Remote Systems</i> -View	370
8.5	Der <i>Remote Monitor</i> -View	371
8.6	Eine neue Verbindung anlegen	371
8.7	Die Verbindung aktivieren	374
8.8	Durch Ressourcen navigieren	374
8.9	Filter und Filterpools erzeugen	375
8.10	Profile verwalten	376
8.11	Ein Projekt auf den entfernten Rechner exportieren	377
8.12	Entfernte Projekte	377
8.13	DataStore	378
8.13.1	DataStore auf Linux einrichten	378
8.13.2	DataStore auf Windows einrichten	379
8.14	Den entfernten Rechner (fern-)steuern	379
8.14.1	Kommandos aufrufen	379
8.14.2	Terminals	379

8.15	Programme auf dem entfernten Rechner ausführen	380
8.15.1	Programme auf entfernten Rechnern starten	381
8.15.2	Beispiel: Cross-Development für Raspberry Pi	381
8.15.3	Projekt auf dem Pi über Eclipse debuggen	385
9	Autotools	387
9.1	Autotools-Plugin installieren	387
9.2	Autotools-Projekt erzeugen	387
9.3	Editoren für Autotools-Dateien	388
9.4	Autotools-Projekte übersetzen	389
9.5	Autotools manuell aufrufen	389
9.6	Globale Voreinstellungen	390
9.7	Projekteinstellungen	390
10	Linux-Tools	391
10.1	Linux-Tools installieren	391
10.2	Valgrind-Plugin	392
10.2.1	Hintergründe	392
10.2.2	Erste Schritte	393
10.3	Profiling-Kategorien	396
10.4	Andere Valgrind-Tools.....	397
10.4.1	Valgrind-Tools konfigurieren	397
10.4.2	Heap mit <i>massif</i> analysieren	398
10.4.3	Cache-Zugriffe mit <i>cachegrind</i> ermitteln.....	400
10.5	Profiling mit GProf	402
10.6	Docker Tooling	403
10.6.1	Verbindung zur Docker-Instanz einrichten	404
10.6.2	Docker-Images.....	405
10.6.3	Programm auf Docker-Container ausführen	406
10.7	Weitere Plugins	407
Anhang		409
Literaturverzeichnis		411
Glossar.....		413
Index		415