

carsten SEIFERT

Komplett in
FARBE

SPIELE ENTWICKELN MIT Unity

3D-GAMES MIT **UNITY** UND **C#** FÜR
DESKTOP, WEB & MOBILE



Auf DVD: komplettes Spiel
plus Videotutorials

HANSER

Hinweis:

Zu diesem Buch gehört eine DVD.
Sollte diese DVD nicht beiliegen, können Sie
sie unter fachbuch@hanser.de kostenlos
anfordern.

Bleiben Sie auf dem Laufenden!



Unser **Computerbuch-Newsletter** informiert Sie monatlich über neue Bücher und Termine. Profitieren Sie auch von Gewinnspielen und exklusiven Leseproben. Gleich anmelden unter



www.hanser-fachbuch.de/newsletter



Hanser Update ist der IT-Blog des Hanser Verlags mit Beiträgen und Praxistipps von unseren Autoren rund um die Themen Online Marketing, Webentwicklung, Programmierung, Softwareentwicklung sowie IT- und Projektmanagement. Lesen Sie mit und abonnieren Sie unsere News unter



www.hanser-fachbuch.de/update



Carsten Seifert

Spiele entwickeln mit Unity

3D-Games mit Unity und C#
für Desktop, Web & Mobile

HANSER

**»Der Weltuntergang steht bevor,
aber nicht so, wie Sie denken.
Dieser Krieg jagt nicht alles in die Luft,
sondern schaltet alles ab.«**



Tom DeMarco
Als auf der Welt das Licht ausging

ca. 560 Seiten. Hardcover
ca. € 19,99 [D] / € 20,60 [A] / sFr 28,90
ISBN 978-3-446-43960-3
Erscheint im November 2014

**Hier klicken zur
Leseprobe**

Sie möchten mehr über Tom DeMarco und seine Bücher erfahren.
Einfach reinklicken unter www.hanser-fachbuch.de/special/demarco

Der Autor:

Carsten Seifert, Süderbrarup

www.hummelwalker.de

Alle in diesem Buch enthaltenen Informationen, Verfahren und Darstellungen wurden nach bestem Wissen zusammengestellt und mit Sorgfalt getestet. Dennoch sind Fehler nicht ganz auszuschließen. Aus diesem Grund sind die im vorliegenden Buch enthaltenen Informationen mit keiner Verpflichtung oder Garantie irgendeiner Art verbunden. Autor und Verlag übernehmen infolgedessen keine juristische Verantwortung und werden keine daraus folgende oder sonstige Haftung übernehmen, die auf irgendeine Art aus der Benutzung dieser Informationen – oder Teilen davon – entsteht.

Ebenso übernehmen Autor und Verlag keine Gewähr dafür, dass beschriebene Verfahren usw. frei von Schutzrechten Dritter sind. Die Wiedergabe von Gebrauchsnamen, Handelsnamen, Warenbezeichnungen usw. in diesem Buch berechtigt deshalb auch ohne besondere Kennzeichnung nicht zu der Annahme, dass solche Namen im Sinne der Warenzeichen- und Markenschutz-Gesetzgebung als frei zu betrachten wären und daher von jedermann benutzt werden dürften.



Bibliografische Information der Deutschen Nationalbibliothek:

Die Deutsche Nationalbibliothek verzeichnet diese Publikation in der Deutschen Nationalbibliografie; detaillierte bibliografische Daten sind im Internet über <http://dnb.d-nb.de> abrufbar.

Dieses Werk ist urheberrechtlich geschützt.

Alle Rechte, auch die der Übersetzung, des Nachdruckes und der Vervielfältigung des Buches, oder Teilen daraus, vorbehalten. Kein Teil des Werkes darf ohne schriftliche Genehmigung des Verlages in irgendeiner Form (Fotokopie, Mikrofilm oder ein anderes Verfahren) – auch nicht für Zwecke der Unterrichtsgestaltung – reproduziert oder unter Verwendung elektronischer Systeme verarbeitet, vervielfältigt oder verbreitet werden.

© 2014 Carl Hanser Verlag München, www.hanser-fachbuch.de

Lektorat: Sieglinde Schärl

Copy editing: Sandra Gottmann, Münster-Nienberge

Herstellung: Irene Weilhart

Erstellung der Dateien für das Beispiel-Game: Alexej Bodemer, Stuhr, www.alexejbodemer.de

Umschlagdesign: Marc Müller-Bremer, München, www.rebranding.de

Umschlagrealisation: Stephan Rönigk

Gesamtherstellung: Kösel, Krugzell

Printed in Germany

Print-ISBN: 978-3-446-43939-9

E-Book-ISBN: 978-3-446-44129-3

Inhalt

Vorwort	XVII
1 Einleitung	1
1.1 Multiplattform-Publishing	1
1.2 Das kann Unity (nicht)	2
1.3 Lizenzmodelle	2
1.4 Aufbau und Ziel des Buches	3
1.5 Weiterentwicklung von Unity	4
1.6 Online-Zusatzmaterial	4
2 Grundlagen	5
2.1 Installation	5
2.2 Oberfläche	5
2.2.1 Hauptmenü	7
2.2.2 Scene View	8
2.2.3 Game View	10
2.2.4 Toolbar	11
2.2.5 Hierarchy	13
2.2.6 Inspector	15
2.2.7 Project Browser	18
2.2.8 Console	20
2.3 Das Unity-Projekt	20
2.3.1 Neues Projekt anlegen	21
2.3.2 Bestehendes Projekt öffnen	22
2.3.3 Projektdateien	22
2.3.4 Szene	23
2.3.5 Game Objects	24
2.3.6 Components	24
2.3.7 Tags	25
2.3.8 Layer	26

2.3.9 Assets	26
2.3.10 Frames	30
2.4 Das erste Übungsprojekt	30
3 C# und Unity	33
3.1 Die Sprache C#	33
3.2 Syntax	34
3.3 Kommentare	35
3.4 Variablen	35
3.4.1 Namenskonventionen	35
3.4.2 Datentypen	36
3.4.3 Schlüsselwort var	37
3.4.4 Datenfelder/Array	37
3.5 Konstanten	39
3.5.1 Enumeration	39
3.6 Typkonvertierung	40
3.7 Rechnen	40
3.8 Verzweigungen	41
3.8.1 if-Anweisungen	41
3.8.2 switch-Anweisung	44
3.9 Schleifen	45
3.9.1 for-Schleife	45
3.9.2 Foreach-Schleife	46
3.9.3 while-Schleife	46
3.9.4 do-Schleife	46
3.10 Klassen	47
3.10.1 Komponenten per Code zuweisen	48
3.10.2 Instanziierung von Nichtkomponenten	48
3.11 Methoden/Funktionen	48
3.11.1 Werttypen und Referenztypen	50
3.11.2 Überladene Methoden	50
3.12 Lokale und globale Variablen	51
3.12.1 Namensverwechslung verhindern mit this	51
3.13 Zugriff und Sichtbarkeit	51
3.14 Statische Klassen und Klassenmember	52
3.15 Parametermodifizierer out/ref	53
3.16 Array-Übergabe mit params	54
3.17 Eigenschaften und Eigenschaftsmethoden	55
3.18 Vererbung	56
3.18.1 Basisklasse und abgeleitete Klassen	56
3.18.2 Vererbung und die Sichtbarkeit	57
3.18.3 Geerbte Methode überschreiben	57

3.18.4	Zugriff auf die Basisklasse	58
3.18.5	Klassen versiegeln	58
3.19	Polymorphie	59
3.20	Schnittstellen	59
3.20.1	Schnittstelle definieren	59
3.20.2	Schnittstellen implementieren	60
3.20.3	Zugriff über eine Schnittstellen	61
3.21	Namespaces	61
3.22	Generische Klassen und Methoden	62
3.22.1	List	63
3.22.2	Dictionary	63
4	Skript-Programmierung	65
4.1	MonoDevelop	65
4.1.1	Hilfe in MonoDevelop	66
4.1.2	Syntaxfehler	66
4.2	Nutzbare Programmiersprachen	67
4.2.1	Warum C#?	67
4.3	Unitys Vererbungsstruktur	68
4.3.1	Object	69
4.3.2	GameObject	69
4.3.3	ScriptableObject	69
4.3.4	Component	69
4.3.5	Transform	70
4.3.6	Behaviour	70
4.3.7	MonoBehaviour	70
4.4	Skripte erstellen	70
4.4.1	Skripte umbenennen	71
4.5	Das Skript-Grundgerüst	71
4.6	Unitys Event-Methoden	72
4.6.1	Update	72
4.6.2	FixedUpdate	73
4.6.3	Awake	73
4.6.4	Start	74
4.6.5	OnGUI	74
4.6.6	LateUpdate	75
4.7	Komponentenprogrammierung	75
4.7.1	Auf GameObjects zugreifen	76
4.7.2	GameObjects aktivieren und deaktivieren	77
4.7.3	GameObjects zerstören	78
4.7.4	GameObjects erstellen	78
4.7.5	Auf Components zugreifen	78
4.7.6	Components hinzufügen	80

4.7.7	Components entfernen	81
4.7.8	Components aktivieren und deaktivieren	81
4.8	Zufallswerte	81
4.9	Parallel Code ausführen	82
4.9.1	WaitForSeconds	83
4.10	Verzögerte und wiederholende Funktionsaufrufe mit Invoke	84
4.10.1	Invoke	84
4.10.2	InvokeRepeating, IsInvoking und CancelInvoke	84
4.11	Daten speichern und laden	85
4.11.1	PlayerPrefs-Voreinstellungen	85
4.11.2	Daten speichern	86
4.11.3	Daten laden	86
4.11.4	Key überprüfen	87
4.11.5	Löschen	87
4.11.6	Save	87
4.12	Szeneübergreifende Daten	88
4.12.1	Werteübergabe mit PlayerPrefs	88
4.12.2	Zerstörung unterbinden	89
4.13	Debug-Klasse	91
4.14	Kompilierungsreihenfolge	92
4.14.1	Programmsprachen mischen und der sprachübergreifende Zugriff	92
4.15	Ausführungsreihenfolge	93
5	Objekte in der dritten Dimension	95
5.1	Das 3D-Koordinatensystem	95
5.2	Vektoren	96
5.2.1	Ort, Winkel und Länge	97
5.2.2	Normalisieren	98
5.3	Das Mesh	98
5.3.1	Normalenvektor	99
5.3.2	MeshFilter und MeshRenderer	100
5.4	Transform	101
5.4.1	Kontextmenü der Transform-Komponente	101
5.4.2	Objekthierarchien	102
5.4.3	Scripting mit Transform	103
5.4.4	Quaternion	104
5.5	Shader und Materials	104
5.5.1	Material-Eigenschaften	105
5.5.2	Neues Material erstellen	108
5.5.3	Normalmaps erstellen	108
5.5.4	UV Mapping	110
5.6	3D-Modelle einer Szene zufügen	111

5.6.1	Primitives	111
5.6.2	3D-Modelle importieren	112
5.6.3	In Unity modellieren	113
5.6.4	Prozedurale Mesh-Generierung	114
6	Kameras, die Augen des Spielers	115
6.1	Die Kamera	115
6.1.1	Komponenten eines Kamera-Objektes	117
6.2	Kamerasteuerung	117
6.2.1	Statische Kamera	117
6.2.2	Parenting-Kamera	118
6.2.3	Kamera-Skripte	118
6.3	ScreenPointToRay	120
6.4	Mehrere Kameras	121
6.4.1	Kamerawechsel	121
6.4.2	Split-Screen	122
6.4.3	Einfache Minimap	123
6.4.4	Render Texture	125
6.5	Image Effects	125
6.6	Skybox	127
6.6.1	Skybox selber erstellen	127
7	Licht und Schatten	129
7.1	Ambient Light	129
7.2	Lichtarten	130
7.2.1	Directional Light	131
7.2.2	Point Light	132
7.2.3	Spot Light	132
7.2.4	Area Light	133
7.3	Schatten	134
7.3.1	Einfluss des MeshRenderers auf Schatten	135
7.4	Light Cookies	135
7.4.1	Import Settings eines Light Cookies	136
7.4.2	Light Cookies und Point Lights	136
7.5	Light Halos	138
7.5.1	Unabhängige Halos	138
7.6	Lens Flares	139
7.6.1	Eigene Flares	139
7.7	Projector	140
7.7.1	Standard Projectors	140
7.8	Lightmapping	141
7.9	Rendering Paths	144
7.9.1	Forward Rendering	144

7.9.2	Vertex Lit	145
7.9.3	Deferred Lighting	145
8	Physik in Unity	147
8.1	Physikberechnung	147
8.2	Rigidbody	148
8.2.1	Rigidbody kennenlernen	149
8.2.2	Masseschwerpunkt	150
8.2.3	Kräfte und Drehmomente zufügen	151
8.3	Kollisionen	154
8.3.1	Collider	154
8.3.2	Trigger	158
8.3.3	Static Collider	159
8.3.4	Kollisionen mit schnellen Objekten	159
8.3.5	Terrain Collider	161
8.3.6	Layer-basierende Kollisionserkennung	161
8.3.7	Mit Layer-Masken arbeiten	161
8.4	Wheel Collider	163
8.4.1	Wheel Friction Curve	164
8.4.2	Entwicklung einer Fahrzeugsteuerung	166
8.4.3	Autokonfiguration	172
8.4.4	Fahrzeugstabilität	174
8.5	Physics Materials	175
8.6	Joints	176
8.6.1	Fixed Joint	176
8.6.2	Spring Joint	176
8.6.3	Hinge Joint	177
8.7	Raycasting	177
8.8	Character Controller	178
8.8.1	SimpleMove	179
8.8.2	Move	180
8.8.3	Kräfte zufügen	181
8.8.4	Einfacher First Person Controller	181
9	Maus, Tastatur, Touch	185
9.1	Virtuelle Achsen und Tasten	185
9.1.1	Der Input-Manager	185
9.1.2	Virtuelle Achsen	187
9.1.3	Virtuelle Tasten	187
9.1.4	Steuern mit Mauseingaben	188
9.1.5	Joystick-Inputs	188
9.1.6	Anlegen neuer Inputs	189
9.2	Achsen- und Tasteneingaben auswerten	189

9.2.1	GetAxis	189
9.2.2	GetButton	190
9.3	Tastatureingaben auswerten	190
9.3.1	GetKey	191
9.3.2	anyKey	191
9.4	Mauseingaben auswerten	192
9.4.1	GetMouseButton	192
9.4.2	mousePosition	192
9.4.3	Mauszeiger ändern	193
9.5	Touch-Eingaben auswerten	194
9.5.1	Der Touch-Typ	195
9.5.2	Input.touches	195
9.5.3	TouchCount	196
9.5.4	GetTouch	196
9.5.5	Touch-Controls	196
9.6	Beschleunigungssensor auswerten	197
9.6.1	Input.acceleration	198
9.6.2	Tiefpass-Filter	199
9.7	Steuerungen bei Mehrspieler-Games	199
9.7.1	Split-Screen-Steuerung	200
9.7.2	Netzwerkspiele	200
10	Audio	203
10.1	AudioListener	203
10.2	AudioSource	204
10.2.1	Durch Mauern hören verhindern	206
10.2.2	Sound starten und stoppen	207
10.2.3	Temporäre AudioSource	208
10.3	AudioClip	209
10.3.1	3D-Sound und Hintergrundmusik	209
10.3.2	Länge ermitteln	209
10.4	Reverb Zone	210
10.5	Filter	211
11	Partikeleffekte mit Shuriken	213
11.1	Editor-Fenster	214
11.2	Particle Effect Control	215
11.3	Numerische Parametervarianten	215
11.4	Farbparameter-Varianten	216
11.5	Default-Modul	216
11.6	Effekt-Module	217
11.6.1	Emission	218
11.6.2	Shape	218

11.6.3	Velocity over Lifetime	219
11.6.4	Limit Velocity over Lifetime	219
11.6.5	Force over Lifetime	220
11.6.6	Color over Lifetime	220
11.6.7	Color by Speed	220
11.6.8	Size over Lifetime	220
11.6.9	Size by Speed	221
11.6.10	Rotation over Lifetime	221
11.6.11	Rotation by Speed	221
11.6.12	External Forces	221
11.6.13	Collision	221
11.6.14	Sub Emitter	223
11.6.15	Texture-Sheet-Animation	223
11.6.16	Renderer	224
11.7	Partikelemission starten, stoppen und unterbrechen	226
11.7.1	Play	226
11.7.2	Stop	226
11.7.3	Pause	227
11.7.4	enableEmission	227
11.8	OnParticleCollision	227
11.8.1	GetCollisionEvents	227
11.9	Feuer erstellen	228
11.9.1	Materials erstellen	229
11.9.2	Feuer-Partikelsystem	229
11.9.3	Rauch-Partikelsystem	232
11.10	Wassertropfen erstellen	236
11.10.1	Tropfen-Material erstellen	236
11.10.2	Wassertropfen-Partikelsystem	236
11.10.3	Kollisionspartikelsystem	239
11.10.4	Kollisionsound	241
12	Landschaften gestalten	243
12.1	Was Terrains können und wo die Grenzen liegen	244
12.2	Terrainhöhe verändern	244
12.2.1	Pinsel	245
12.2.2	Oberflächen anheben und senken	245
12.2.3	Plateaus und Schluchten erstellen	246
12.2.4	Oberflächen weicher machen	247
12.2.5	Heightmaps	247
12.3	Terrain texturieren	249
12.3.1	Textur-Pinsel	250
12.3.2	Texturen verwalten	250
12.4	Bäume und Sträucher	252
12.4.1	Bedienung des Place Tree-Tools	252

12.4.2	Wälder erstellen	253
12.4.3	Mit Bäumen kollidieren	253
12.5	Gräser und Details hinzufügen	254
12.5.1	Detail-Meshs	254
12.5.2	Gräser	255
12.5.3	Quelldaten nachladen	256
12.6	Terrain-Einstellungen	256
12.6.1	Base Terrain	257
12.6.2	Resolution	257
12.6.3	Tree & Details Objects	258
12.6.4	Wind Settings	258
12.6.5	Zur Laufzeit Terrain-Eigenschaften verändern	259
12.7	Der Weg zum perfekten Terrain	260
12.8	Gewässer	261
13	Wind Zones	263
13.1	Spherical vs. Directional	264
13.2	Wind Zone - Eigenschaften	265
13.3	Frische Brise	266
13.4	Turbine	266
14	GUI	267
14.1	GUIElements	268
14.1.1	GUIElements ausrichten	268
14.1.2	GUIText positionieren	269
14.1.3	GUITexture skalieren und positionieren	269
14.1.4	Interaktivität	270
14.2	3DText	271
14.3	OnGUI-Programmierung	272
14.3.1	GUI	273
14.3.2	GUILayout	275
14.3.3	GUIStyle und GUISkin	276
14.4	uGUI	278
14.4.1	Canvas	278
14.4.2	RectTransform	281
14.4.3	uGUI-Sprite Import	285
14.4.4	Grafische Controls	286
14.4.5	Interaktive Controls	289
14.4.6	Controls designen	294
14.4.7	Animationen in uGUI	295
14.4.8	Event Trigger	296
14.5	Screen-Klasse	297
14.5.1	Schriftgröße dem Bildschirm anpassen	298

15	Prefabs	299
15.1	Prefabs erstellen und nutzen	299
15.2	Prefab-Instanzen erzeugen	299
15.2.1	Instanzen per Code erstellen	300
15.2.2	Instanzen weiter bearbeiten	300
15.3	Prefabs ersetzen und zurücksetzen	301
16	Internet und Datenbanken	303
16.1	Die WWW-Klasse	303
16.1.1	Rückgabewert-Formate	304
16.1.2	Parameter übergeben	305
16.2	Datenbank-Kommunikation	306
16.2.1	Daten in einer Datenbank speichern	306
16.2.2	Daten von einer Datenbank abfragen	307
16.2.3	Rückgabewerte parsen	309
16.2.4	Datenhaltung in eigenen Datentypen	310
16.2.5	HighscoreCommunication.cs	312
16.2.6	Datenbankverbindung in PHP	313
17	Animationen	315
17.1	Allgemeiner Animation-Workflow	316
17.2	Animationen erstellen	316
17.2.1	Animation View	317
17.2.2	Curves vs. Dope Sheet	318
17.2.3	Animationsaufnahme	318
17.2.4	Beispiel Fallgatter-Animation	322
17.3	Animationen importieren	323
17.3.1	Rig	323
17.3.2	Animationen	326
17.4	Animationen einbinden	329
17.4.1	Animator Controller	329
17.4.2	Animator-Komponente	334
17.4.3	Beispiel Fallgatter Animator Controller	335
17.5	Controller-Skripte	337
17.5.1	Parameter des Animator Controllers setzen	337
17.5.2	Animation States abfragen	338
17.5.3	Beispiel Fallgatter Controller-Skript	339
17.6	Animation Events	341
17.6.1	Beispiel Fallgatter-Bewegung	342
18	Künstliche Intelligenz	345
18.1	NavMeshAgent	346
18.1.1	Eigenschaften der Navigationskomponente	346
18.1.2	Zielpunkt zuweisen	347

18.2	NavigationMesh	347
18.2.1	Object Tab	349
18.2.2	Bake Tab	349
18.2.3	Layers Tab	350
18.3	NavMeshObstacle	350
18.4	Point & Click-Steuerung für Maus und Touch	351
19	Fehlersuche und Performance	353
19.1	Fehlersuche	353
19.1.1	Breakpoints	354
19.1.2	Variablen beobachten	355
19.1.3	Console Tab nutzen	355
19.2	Performance	356
19.2.1	Rendering-Statistik	356
19.2.2	Analyse mit dem Profiler	358
19.2.3	Echtzeit-Analyse auf Endgeräten	359
20	Spiele erstellen und publizieren	361
20.1	Der Build-Prozess	361
20.1.1	Szenen des Spiels	362
20.1.2	Plattformen	363
20.1.3	Notwendige SDKs	363
20.1.4	Plattformspezifische Optionen	364
20.1.5	Developer Builds	364
20.2	Publizieren	365
20.2.1	App	365
20.2.2	Browser-Game	366
20.2.3	Desktop-Anwendung	366
21	Beispiel-Game	369
21.1	Level-Design	370
21.1.1	Modellimport	371
21.1.2	Materials zuweisen	372
21.1.3	Prefabs erstellen	372
21.1.4	Dungeon erstellen	375
21.1.5	Dekoration erstellen	378
21.2	Inventarsystem erstellen	380
21.2.1	Verwaltungslogik	380
21.2.2	Oberfläche des Inventarsystems	386
21.2.3	Inventar-Items	388
21.3	Game Controller	393
21.4	Spieler erstellen	393
21.4.1	Lebensverwaltung	395
21.4.2	Spielersteuerung	403
21.4.3	Wurfstein entwickeln	411

21.5	Quest erstellen	416
21.5.1	Erfahrungspunkte verwalten	416
21.5.2	Questgeber erstellen	418
21.5.3	Sub-Quest erstellen	425
21.6	Gegner erstellen	430
21.6.1	Model-, Rig- und Animationsimport	430
21.6.2	Komponenten und Prefab konfigurieren	431
21.6.3	Animator Controller erstellen	432
21.6.4	NavMesh erstellen	434
21.6.5	Umgebung und Feinde erkennen	435
21.6.6	Gesundheitszustand verwalten	438
21.6.7	Künstliche Intelligenz entwickeln	442
21.7	Eröffnungsszene	449
21.7.1	Startszene erstellen	449
21.7.2	Startmenü erstellen	450
21.8	Web-Player-Anpassungen	454
21.8.1	Web-Player-Template ändern	454
21.8.2	Quit-Methode im Web-Player abfangen	455
21.9	Umstellung auf uGUI	455
21.9.1	Skriptanpassungen	456
21.9.2	Controls erstellen	459
21.10	So könnte es weitergehen	466
22	Der Produktionsprozess in der Spieleentwicklung	469
22.1	Die Produktionsphasen	469
22.1.1	Ideen- und Konzeptionsphase	470
22.1.2	Planungsphase	470
22.1.3	Entwicklungsphase	470
22.1.4	Testphase	471
22.1.5	Veröffentlichung und Postproduktion	471
22.2	Das Game-Design-Dokument	471
	Schlusswort	473
	Index	475

Vorwort

Für viele von uns sind Computerspiele heutzutage allgegenwärtige Wegbegleiter. Egal wo, auf dem Smartphone, dem Tablet oder dem heimischen PC sind sie installiert und täglich in Benutzung. Manchmal dienen sie als Zeitvertreib, bis der nächste Bus kommt, manchmal sind sie aber auch Bestandteil eines intensiven Hobbys.

Aber nicht nur das Spielen kann Spaß machen, auch das Entwickeln dieser Games kann begeistern. Sowohl im Freizeitbereich als auch in der Arbeitswelt wird der Beruf des Spieleentwicklers immer beliebter. Es ist also kein Wunder, dass sich in den letzten Jahren bereits ganze Studiengänge dem Entwickeln von Computerspielen gewidmet haben.

In diesem Buch möchte ich Ihnen Unity, eine weit verbreitete Entwicklungsumgebung für Computerspiele, näherbringen und erläutern, wie Sie mit diesem Werkzeug Spiele selber entwickeln können. Dabei richtet sich das Buch sowohl an Einsteiger, Umsteiger und auch an Spieleentwickler, die mit Unity nun richtig durchstarten möchten.

An dieser Stelle möchte ich mich ganz besonders bei meiner Frau Cornelia bedanken, die mich während des Schreibens so geduldig unterstützt hat und mir jederzeit beim Formulieren und Korrigieren hilfsbereit zur Seite stand.

Auch danke ich ganz herzlich Alexej Bodemer, der für das Beispiel-Game dieses Buches alle 3D-Modelle, Texturen und Musikdateien entworfen und zur Verfügung gestellt hat.

Zudem gilt mein Dank Will Goldstone und Unity Technologies, die mir die bis dato aktuellsten Beta-Versionen zur Verfügung gestellt haben.

Weiter möchte ich Frau Sieglinde Schär, Kristin Rothe und dem gesamten Hanser-Verlag-Team danken, die mir nicht nur das Schreiben dieses Buches ermöglicht haben, sondern auch jederzeit mit Rat und Tat zur Seite standen.

Nicht zuletzt danke ich auch meiner gesamten Community, die mich während des Schreibens auf meinem Blog und meinen sozialen Kanälen so konstruktiv begleitet hat.

Süderbrarup, August 2014

Carsten Seifert

1

Einleitung

Computerspiele gehören heutzutage zu den beliebtesten Freizeitgestaltungen unserer Zeit. Mit Zunahme der Popularität ist aber auch der Anspruch an diese Spiele gestiegen. Während früher ein einzelner Programmierer ausreichte, um alle notwendigen Aufgaben zu erledigen, werden heutzutage anspruchsvolle Computerspiele meist von Teams, bestehend aus Modellierern, Grafikern, Sounddesignern, Level-Designern und natürlich auch Programmierern, umgesetzt.

Um den stetig wachsenden Ansprüchen zu genügen, sind aber auch die Werkzeuge der Entwickler ständig mitgewachsen. Eines dieser Werkzeuge ist Unity. Unity ist eine Spieleentwicklungsumgebung für Windows- und Mac OS X-Systeme und wird von der aus Dänemark stammenden Firma Unity Technologies entwickelt. Mit ihr können Sie sowohl interaktive 3D- als auch 2D-Inhalte erstellen. Ich spreche deshalb von Inhalten und nicht nur von Spielen, weil Unity zwar eigentlich für die Entwicklung von 3D-Spielen gedacht war, mittlerweile aber auch immer häufiger Anwendung in anderen Bereichen findet. So wird es beispielsweise für Architekturvisualisierungen genutzt, im E-Learning-Bereich eingesetzt oder in der Digital-Signage-Branche für das Erstellen digitaler Werbe- und Informationssysteme genommen.

Da Unity ursprünglich für die Entwicklung von 3D-Spielen konzipiert wurde, lautet die Internet-Adresse der Firma *unity3d.com*. Dies ist der Grund, weshalb die Entwicklungssoftware auch gerne mal „Unity3D“ genannt wird, was aber eben nicht ganz korrekt ist.

■ 1.1 Multiplattform-Publishing

Eine besondere Stärke von Unity ist die Unterstützung von Multiplattform-Publishing. Das bedeutet, dass Sie in Unity ein Spiel einmal entwickeln können, das Sie dann aber für mehrere Plattformen exportieren können. Aktuell werden standardmäßig Windows Desktop-Programme, OS X, Linux, Windows Store Apps, iOS, Android, Windows Phone 8, Blackberry 10, Google Native Client sowie ein eigener Webplayer als Publishing-Format unterstützt.

Der erwähnte Webplayer, auch Unity-PlugIn genannt, ähnelt dabei dem Flash Player und wird im Browser installiert. Mit ihm können Spiele, die mit Unity entwickelt wurden, direkt im Browser ohne große funktionale Einbußen gespielt werden.

Sie können mit Unity auch für die Konsolen Xbox, Wii und PlayStation entwickeln. Dafür müssen Sie allerdings extra Lizenzen erwerben, die zum einen nicht günstig und zum anderen nur für Firmen verfügbar sind, die von den jeweiligen Konsolenherstellern auch als Entwickler akzeptiert wurden. Deshalb werde ich die Konsolenentwicklung in diesem Buch auch außen vor lassen und nicht näher beleuchten.

■ 1.2 Das kann Unity (nicht)

Unity bringt eine ganze Reihe an nützlichen Werkzeugen mit, um Spiele und andere 3D-Anwendungen zu entwickeln. So gibt es neben einer ausgeklügelten Physik-Engine auch Tools für Partikeleffekte, zur Landschaftsgestaltung oder auch für Animationen. Außerdem wird Unity mit einer extra angepassten Version der Softwareentwicklungsumgebung Mono-Develop ausgeliefert, in der die Programmierung umgesetzt und das Debugging vorgenommen werden kann.

Eines ist Unity allerdings nicht: Es ist keine 3D-Modellierungssoftware. Unity bietet zwar von Haus aus einige 3D-Grundobjekte an, sogenannte Primitives, die für kleinere Aufgaben genutzt werden können, aber für richtige Modellierungsaufgaben sollten Sie auf die dafür entwickelten Spezialtools wie 3ds Max oder das kostenlose Blender zurückgreifen.

■ 1.3 Lizenzmodelle

Die Spieleentwicklungsumgebung gibt es in einer kostenlosen Ausführung (auch Free, Indie oder Unity Basic genannt) und in einer kostenpflichtigen Edition. Die kostenlose Version unterstützt bereits alle oben genannten Zielplattformen und erlaubt eine komplette Spieleentwicklung bis hin zum fertigen Spiel. Die kostenpflichtige Version, die auch Unity Pro genannt wird, beinhaltet dann noch weitere Funktionen, um das Spiel grafisch weiter aufzubereiten, die Performance zu optimieren oder Zusatzfunktionen für Animationen und künstliche Intelligenz. Allerdings sind für bestimmte Plattformen wie iOS, Android und BlackBerry 10 noch weitere Lizenzen notwendig, um die zusätzlichen Pro-Features auch dort zu erhalten.

Da sowohl Unity Pro als auch die Indie-Version bzw. Unity Basic kommerziell genutzt werden dürfen, gibt es beim Einsatz der kostenlosen Version die Vorgabe, dass nur Firmen bzw. Entwickler diese einsetzen dürfen, die nicht mehr als 100 000 US-Dollar Umsatz in einem Geschäftsjahr machen.

■ 1.4 Aufbau und Ziel des Buches

Mit diesem Buch werden Sie lernen, auf Basis von Unity eigene 3D-Spiele zu entwickeln. Sie werden sich mit den unterschiedlichen Tools der Game Engine vertraut machen und die Skript-Programmierung erlernen. Dabei steht aber nicht das Ziel im Fokus, wirklich alle Funktionen und Möglichkeiten zu beleuchten, die Unity dem Entwickler anbietet. Vielmehr zeige ich Ihnen, wie die unterschiedlichen Bereiche von Unity funktionieren und miteinander zusammenarbeiten. Denn der Funktionsumfang dieser Spieleentwicklungsumgebung ist mittlerweile so umfangreich geworden, dass es gar nicht mehr möglich ist, alle Tools und deren Möglichkeiten bis ins letzte Detail in einem einzigen Buch ausführlich zu behandeln. Daher liegt der Schwerpunkt auf den Kernfunktionen, die in der kostenlosen Unity-Version für die 3D-Spieleentwicklung bereitgestellt werden, ergänzt um Anwendungsbeispiele und Praxistipps.

Möchten Sie weiter in die Tiefe eines speziellen Tools gehen oder mehr Informationen zu bestimmten Scripting-Klassen erhalten, empfehle ich Ihnen die mit Unity mitgelieferten Hilfe-Dokumente, die Sie über das Help-Menü erreichen. Dort finden Sie sowohl ein ausführliches Manual über die in Unity integrierten Tools sowie eine *Scripting-Referenz* über alle Unity-Klassen und deren Möglichkeiten. Letztere finden Sie auch über die Hilfe von MonoDevelop, der mitgelieferten Programmierumgebung.

Spieleentwicklung wird häufig auch als Spieleprogrammierung bezeichnet, auch wenn viele Aufgaben in der Spieleentwicklung heutzutage nicht mehr programmiert, sondern mithilfe von Tools erledigt werden. Nichtsdestotrotz ist der Programmieranteil bei der Entwicklung eines Spiels doch immer noch sehr hoch. Deshalb wird auch das Buch zunächst mit zwei größeren programmierbezogenen Kapiteln beginnen, wo es unter anderem auch um die Grundlagen der Programmierung geht. Erst danach werden wir in die 3D-Welt von Unity eintauchen und die verschiedenen Werkzeuge behandeln. Der Aufbau ist deshalb so gewählt, weil ich in den folgenden Kapiteln immer wieder kleinere Skript-Beispiele zeige, die Einsatzmöglichkeiten in der Praxis demonstrieren.

Am Ende des Buches werden wir schließlich in einem etwas größeren Kapitel ein Beispiel-Game entwickeln, das viele der behandelten Themen noch einmal aufgreift und die gesamten Zusammenhänge in der Praxis zeigt.



Die DVD zum Buch

Dem Buch liegt eine DVD bei, auf der sich Folgendes befindet:

- Beispiel-Game (Dungeon Crawler) mit allen dazugehörigen Ressourcen
- Anwendungsbeispiel für eine Auto-Steuerung inklusive eines 3D-Modells mit zusätzlichen Skripten
- Beispiel für eine Sprite-Animation
- Vorlage für ein einfaches Übungsprojekt
- ergänzende Video-Tutorials, in denen die Verwendung spezieller Werkzeuge und Verfahren in der Praxis demonstriert wird.

Im Buch werde ich Hinweiskästen, wie den obigen, einsetzen, um Hinweise und Tipps zu geben. Je nach Typ des Hinweises werden diese mit unterschiedlichen Icons ausgezeichnet.



Hinweise zu Inhalten auf der DVD



Hinweise zu weiterführenden Inhalten im Internet



Praxistipps



Allgemeine Hinweise

■ 1.5 Weiterentwicklung von Unity

Die Spieleindustrie gehört zu den Branchen, die sich aktuell am schnellsten verändern. Kein Wunder also, dass auch Unity ständig weiterentwickelt wird und neue Funktionen erhält. Sollten Sie Unterschiede zwischen dem Buch und Ihrer Unity-Version erkennen, wird dies sicher der ständigen Weiterentwicklung von Unity geschuldet sein.

Aber nicht nur der Funktionsumfang wird ständig weiterentwickelt, auch das Lizenzmodell von Unity ist nicht von Veränderungen ausgeschlossen. Vielleicht können Sie schon Funktionen mit Unity Basic nutzen, die in diesem Buch noch als Pro-Feature gelten. Ein gutes Beispiel hierfür sind die Berechtigungen für die Mobile-Entwicklung. Noch vor Kurzem hätten Sie bereits für die Basic-Mobile-Entwicklung extra Lizenzen kaufen müssen. Diese Berechtigungen sind jetzt bereits in der kostenlosen Version integriert. Und auch das Add-On für das Pathfinding (siehe Kapitel „Künstliche Intelligenz“) war anfangs ein Pro-Feature, heutzutage ist es fast komplett in der Indie-Version enthalten.

■ 1.6 Online-Zusatzmaterial

Aufgrund der kontinuierlichen Weiterentwicklung von Unity werde ich Ihnen in einem passwortgeschützten Bereich meiner Website Zusatzmaterialien bereitstellen, die den Inhalt dieses Buches ergänzen und auch aktuell halten sollen.



Passwortgeschützter Bereich für Zusatzmaterialien

URL: <http://www.hummelwalker.de/buch-zusatzmaterial/>

Passwort: **uN1TyZuS4TzMaT3RiAl**

2

Grundlagen

In diesem Kapitel werden Sie die Oberfläche von Unity sowie deren grundlegende Bedienung kennenlernen. Wir werden auf die Struktur eines Unity-Projektes eingehen und die grundlegenden Prinzipien von Unity behandeln.

■ 2.1 Installation

Sollten Sie noch keine Installationsdatei haben, besuchen Sie zunächst die Seite <http://www.unity3d.com> und laden sich dort die aktuelle Unity-Version herunter. Nach dem Download können Sie Unity installieren. Da die Installation eigentlich selbsterklärend ist, sollten Sie lediglich darauf achten, dass Sie eine Komplettinstallation machen und keine Teile davon ausnehmen. Profis können natürlich selber bewerten, was sie benötigen, Anfängern würde ich aber immer eine Komplettinstallation empfehlen.

Wenn Sie die Installation abgeschlossen haben und Unity das erste Mal starten, wird sich der *Project Wizard* von Unity zeigen. Über diesen können Sie ein existierendes Projekt öffnen oder ein neues Projekt erstellen. Mehr zum Öffnen und Erstellen eines Projektes erfahren Sie im Abschnitt „Das Unity-Projekt“. Für den ersten Start können Sie einfach die Standardeinstellungen übernehmen und über **Create** ein neues Projekt erstellen.

■ 2.2 Oberfläche

Die Oberfläche von Unity besteht aus mehreren frei anpassbaren Fenstern (auch Tabs genannt), die sich innerhalb von Unity übereinander und nebeneinander andocken sowie auch außerhalb des Hauptfensters verschieben lassen.

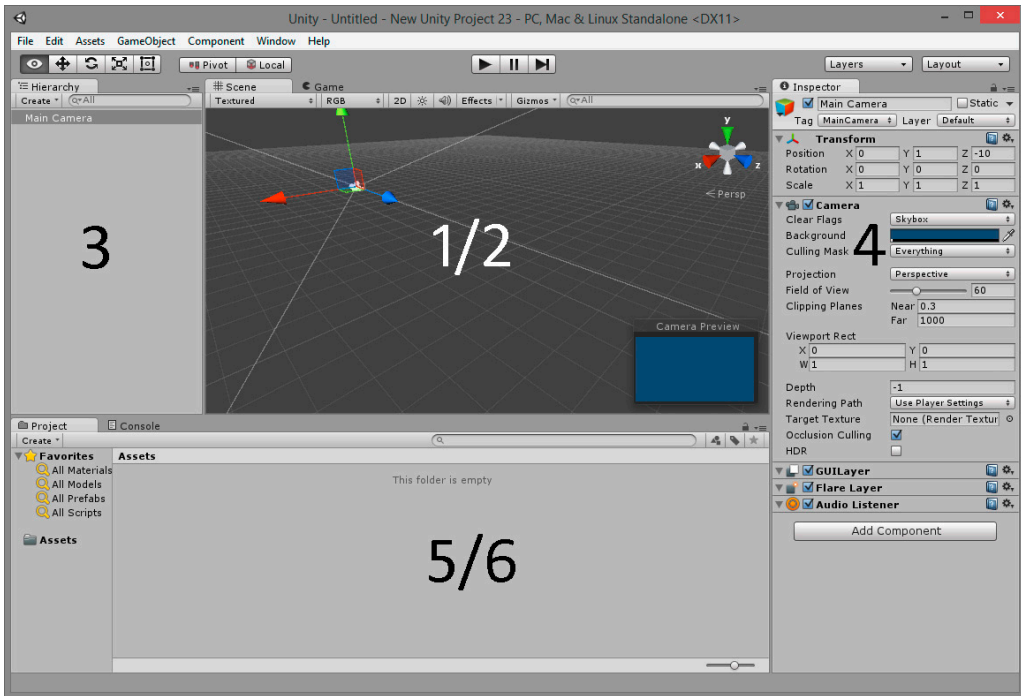


Bild 2.1 Default-Ansicht von Unity

Die Fenster (Tabs) besitzen zwar unterschiedliche Aufgaben, gehören funktional aber alle zusammen und werden deshalb auch kontinuierlich untereinander synchronisiert.

1. **Scene View** dient dem interaktiven Gestalten von Szenen und 3D-Welten.
2. **Game View** dient als Vorschau des fertigen Spiels. Dieses Fenster wird in Bild 2.1 von der *Scene View* verdeckt, wird aber von Unity automatisch nach vorne gebracht, wenn das Spiel gestartet wird.
3. **Hierarchy** zeigt alle in der Szene existierenden Objekte (*GameObjects*) in deren Hierarchiestruktur an.
4. **Inspector** zeigt alle Komponenten und öffentlichen Parameter des aktuell selektierten *GameObjects* an.
5. **Project Browser** dient dem Anzeigen und Verwalten aller digitalen Inhalte (auch *Assets* genannt), die zu dem Projekt gehören.
6. **Console** dient dem Anzeigen von Fehler- und Hinweismeldungen. Dieses Fenster befindet sich in der Default-Ansicht hinter dem *Project Browser*. Um die Meldungen zu sehen, müssen Sie auf den Reiter des Fensters klicken.

Abgesehen von der *Game View* und dem *Console-Tab* können Sie den Tabs nicht nur Informationen entnehmen, sondern auch die Objekte verändern.

Sie können die Tab-Anordnungen in Unity jederzeit speichern und auch zurücksetzen. Hierfür stellt Unity im oberen Hauptmenü über **Window/Layouts** entsprechende Funktionen und Standardanordnungen zur Verfügung. Als Schnellzugriff dient hier das *Layouts-Dropdown-Menü*, das Sie ganz rechts im oberen Bereich von Unity finden.

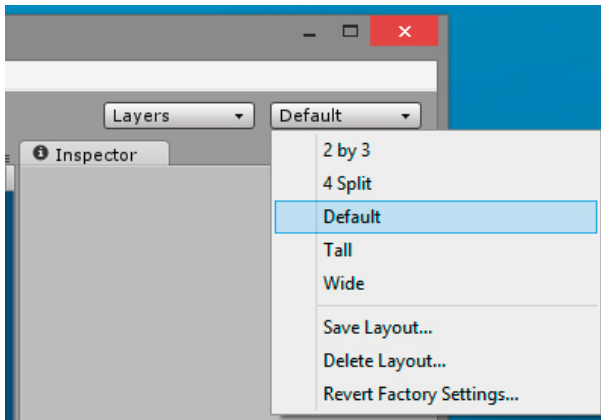


Bild 2.2
Schnellzugriff auf die Layout-Funktionen

Wenn Sie mit **Save Layout** eigene Fenster-Anordnungen speichern, stehen diese Layouts nicht nur in diesem Projekt zur Verfügung, sondern auch in allen anderen Unity-Projekten. Mit **Delete Layout** löschen Sie diese natürlich ebenfalls in allen Projekten. Sollten Sie dabei eine Standardsicht aus Versehen gelöscht haben, können Sie über **Revert Factory Settings** alle Layout-Einstellungen wieder auf die Werkseinstellungen zurücksetzen.

2.2.1 Hauptmenü

Oberhalb aller Subfenster und Toolbars befindet sich das Hauptmenü von Unity. Viele Teile dieses Menüs finden Sie zusätzlich noch einmal als Schnellzugriff-Menüs in anderen Bereichen von Unity wieder, wie z. B. das *Layouts-Drop-down-Menü* (siehe oben).

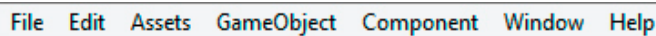


Bild 2.3 Hauptmenü

Hinter diesen Hauptmenüpunkten verbergen sich folgende Funktionen:

- **File** beinhaltet alle Funktionen, die sich mit dem Erstellen und Speichern von Dateien auf Projekt- und Szene-Ebene beschäftigen.
- **Edit** besitzt vor allem Einstellungen zum Verändern von Daten.
- **Asset** beschäftigt sich mit dem Verwalten und Erstellen neuer *Assets*, z. B. von Skripten und Materialien.
- **GameObject** beschäftigt sich vor allem mit dem Erstellen verschiedener, vorkonfigurierter *GameObjects*.
- **Component** bietet alle Standardkomponenten von Unity an.
- **Window** bietet vor allem weitere Fenster/Tabs an, die Sie anzeigen können.
- **Help** besitzt hauptsächlich Quellen und Links zu weiterführenden Informationen für Unity. Hier gelangen Sie auch zum „Unity Manual“ und zu der „Scripting Reference“, wo Sie viele weiterführende Informationen finden.

2.2.2 Scene View

Das Fenster mit der Überschrift „Scene“ wird als *Scene View* bezeichnet und dient dem interaktiven Positionieren und Verändern von Objekten innerhalb einer Szene bzw. eines Levels. Sie können hier neue Objekte hinzufügen, diese verschieben, rotieren und auch skalieren. Ein wichtiges Werkzeug hierfür sind die *Transform-Tools*, die Sie in der Toolbar direkt unter dem Hauptmenü oben links finden. Über Maus und Tastatur können Sie zudem durch die Szene navigieren (siehe Abschnitt 2.2.2.1, „Navigieren in der Scene View“).

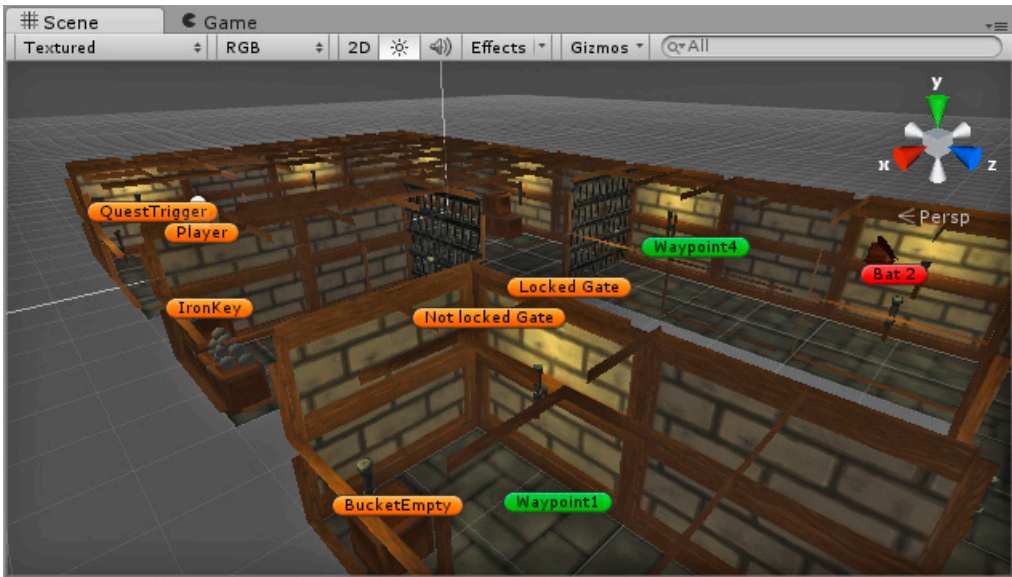


Bild 2.4 Scene View

Möchten Sie ein neues *GameObject* in der *Scene View* platzieren, können Sie einfach ein *GameObject* aus dem *Project Browser* in die *Scene View* hineinziehen und fallen lassen. Anschließend können Sie es noch verschieben und mit den erwähnten *Transform-Tools* modifizieren (siehe Abschnitt 2.2.4, „Toolbar“).

Am oberen Rand der *Scene View* befindet sich eine schmale *Control Bar*, die Ihnen über mehrere Untermenüs verschiedene Einstellmöglichkeiten bietet (siehe Bild 2.4).

- **Mesh Rendering-Menü** wechselt die Ansicht zwischen verschiedenen Darstellungsarten (Texturiert, Drahtgitter usw.).
- **Texture Rendering-Menü** stellt verschiedene Darstellungsformen der Texturen zur Verfügung (RGB, Alpha ...).
- **2D-Button** wechselt zwischen der normalen 3D-Ansicht und einer speziellen Darstellungsform, die besonders für 2D-Games und das GUI-Design geeignet ist. Mehr zu dieser Ansicht erfahren Sie im Abschnitt 2.2.2.2.
- **Beleuchtungs-Button** schaltet die Darstellung der Lichtquellen bzw. deren Auswirkungen in der *Scene View* ein/aus.

- **Sound-Button** schaltet den Ton ein/aus.
- **Effects-Menü** ermöglicht, bestimmte Effekte in der Szene zu aktivieren/deaktivieren (Skybox, Flares ...).
- **Gizmos-Menü** schaltet typbezogen Hilfsgrafiken in der *Scene View* ein/aus (Gizmos sind grafische Hilfsmittel, um nicht sichtbare Objekte wie Kameras, Soundquellen etc. darzustellen).
- **Suchmaske**, um Objekte in der Szene zu suchen (dabei wird alles ausgegraut, außer die gefundenen Objekte)

2.2.2.1 Navigieren in der Scene View

Unity bietet unterschiedliche Möglichkeiten, durch die *Scene View* zu navigieren. Manche Funktionen sind hierbei auch auf unterschiedliche Weise zu erreichen. Im Folgenden möchte ich Ihnen die wichtigsten vorstellen.

- **Objekte selektieren** Sie mit der linken Maustaste.
- **Ansicht in der Höhe und Seite verschieben** Sie mit der mittleren Maustaste bzw. durch Drücken auf das Mausrad. Sie können auch über die Pfeiltasten navigieren oder alternativ auch das Verschiebewerkzeug der *Transform-Tools* aktivieren (Shortcut: `[Q]`) und dann die linke Maustaste nutzen (siehe *Transform-Tools*). Wenn Sie große Szenen besitzen, können Sie die Geschwindigkeit des Verschiebens über das zusätzliche Drücken der `[Umsch]`-Taste (`[Shift]`) noch erhöhen.
- **Ansicht in der Tiefe und Seite verschieben** Sie über die Pfeiltasten. Hierbei bewegen Sie sich den Pfeiltasten entsprechend durch die Szene.
- **Ansicht drehen** Sie über die rechte Maustaste. Alternativ können Sie auch die Kombination aus `[Alt]`-Taste und der linken Maustaste nutzen.
- **Ansicht zoomen** Sie über das Drehen des Mausrads. Alternativ können Sie auch die `[Alt]`-Taste und die rechte Maustaste drücken, während Sie den Mauszeiger bewegen.
- **Objekte fokussieren** Sie, indem Sie ein Objekt in der *Scene View* oder in der *Hierarchy* selektieren und dann die Taste `[F]` drücken. Die Ansicht wird dann so gesetzt, dass das Objekt im Mittelpunkt steht.

2.2.2.2 Scene Gizmo

Oben rechts innerhalb der *Scene View* finden Sie das *Scene Gizmo*, das die Orientierung des Koordinatensystems zur aktuellen Sicht der *Scene View* darstellt. Das *Scene Gizmo* ist dabei wie ein Kompass und zeigt Ihnen, von welcher Seite Sie aktuell die Szene betrachten.

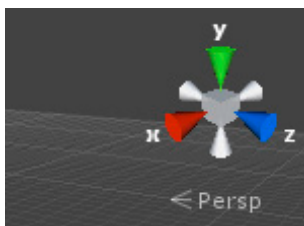


Bild 2.5
Scene Gizmo

Wenn Sie auf die unterschiedlichen Achsen des *Scene Gizmos* klicken, wechselt die Ansicht dabei in die jeweilige Seiten-, Drauf- oder Frontansicht. Ein Klick auf den mittleren Würfel ändert zudem die Ansicht zwischen orthogonal und perspektivisch.

- **Orthogonal** ignoriert Tiefenverhältnisse. So werden beispielsweise zwei gleich große Objekte immer gleich groß dargestellt, egal wie weit diese vom Betrachter entfernt sind.
- **Perspektivisch** stellt Objekte „natürlich“ dar und berücksichtigt die Entfernung vom Betrachter.

Unter dem *Scene Gizmo* wird Ihnen in Textform zusätzlich angezeigt, welche Ansicht Sie aktuell gewählt haben. Die drei parallel verlaufenden Linien bedeuten hierbei orthogonal, die auseinandergehenden Linien bedeuten perspektivisch. Ein Klick auf diesen Text wechselt ebenfalls zwischen diesen beiden Ansichtsformen.

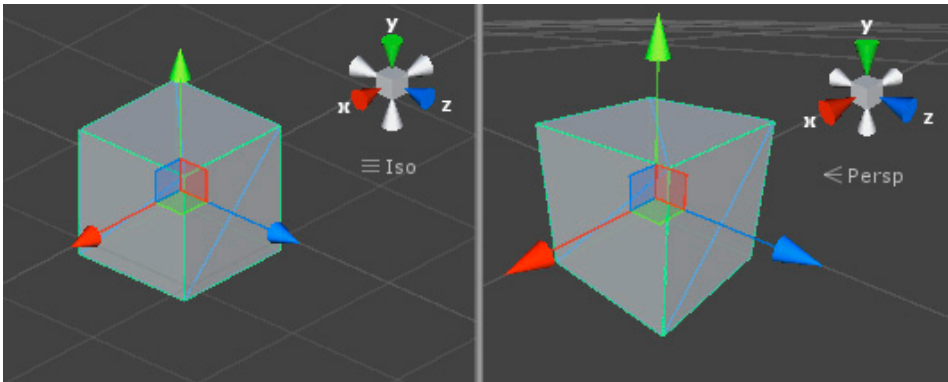


Bild 2.6 Vergleich: Orthogonal (links) vs. Perspektivisch (rechts)



2D-Button

Der *2D-Button*, den Sie in der *Control Bar* der *Scene View* finden, ändert lediglich die Ansicht auf „Orthogonal“ und dreht die räumliche Betrachtung so, dass der Nutzer in Richtung der positiven Z-Achse schaut, während die Y-Achse nach oben zeigt. Er ruft also nur eine Standardansicht auf, die sich für 2D-Spiele als praktisch erwiesen hat. Ansonsten nimmt dieser Button keinen Einfluss auf das Spiel.

2.2.3 Game View

Die *Game View* dient dem Testen Ihres Projektes. Sie zeigt das Spiel aus Sicht der Kamera und ermöglicht Ihnen, das Spiel so zu betrachten, wie es später nach der Erstellung aussehen wird. Wenn Sie das Spiel über die Play-Taste starten, können Sie über dieses Fenster Ihr Spiel testen.