

matthias MELZER

Second Life

PROGRAMMIERUNG



MIT DER

Linden Scripting Language



HANSER



Inhalt

Inhalt	VII
Vorwort	XIII
1 Einführung in LindenScript	1
1.1 Die Entwicklungsumgebung	1
1.2 Hello, Avatar!	3
1.3 Variablen	5
1.3.1 Deklarieren von Variablen	6
1.3.2 Variablen Werte zuweisen	8
1.4 Datentypen.....	9
1.4.1 Integer	9
1.4.2 Float	11
1.4.3 String.....	11
1.4.4 Vector.....	12
1.4.5 Rotation/Quaternion	18
1.4.6 Key.....	19
1.4.7 List	19
1.5 Typumwandlung	21
1.5.1 Explizite Typumwandlung	21
1.5.2 Implizite Typumwandlung	25
1.6 Operatoren	26
1.6.1 Zuweisungsoperator	26
1.6.2 Arithmetische Operatoren	27
1.6.3 Inkrement- und Dekrementoperatoren.....	28
1.6.4 Relationale Operatoren.....	29
1.6.5 Logische Operatoren	31
1.6.6 Binäre Operatoren	33
1.6.7 Sonstige Operatoren	36
1.6.8 Priorität der Operatoren.....	37

1.7	Konstanten.....	38
1.8	Anweisungen und Blöcke.....	39
1.9	Kontrollstrukturen.....	40
1.9.1	if und if-else.....	40
1.9.2	while-Schleife.....	43
1.9.3	do-while-Schleife.....	44
1.9.4	for-Schleife.....	44
1.9.5	jump.....	46
1.10	Funktionen.....	48
1.11	In LindenScript integrierte Funktionen.....	52
1.11.1	Funktionen zur string-Bearbeitung.....	52
1.11.2	Funktionen zur list-Bearbeitung.....	56
1.12	In Abschnitte eingeteilte Listen (Strided Lists).....	67
1.13	Kartenziehen – ein erstes Beispielprogramm.....	69
1.14	Ereignisse/Events.....	75
1.14.1	Dem Benutzer zuhören – der Event listen.....	76
1.14.2	Zeitgesteuerte Skripte – der Event timer.....	79
1.14.3	Anweisungen beim Rezzen eines Objektes ausführen – der Event on_rez.....	80
1.15	Zustände/States.....	82
2	Primitives/Objekte.....	87
2.1	Was ist ein Primitive? Was ist ein Objekt?.....	87
2.1.1	Primitives.....	87
2.1.2	Objekte.....	88
2.2	Identität eines Prims.....	89
2.3	Zugriffsrechte.....	90
2.4	Einfache Manipulationen von Prim-Eigenschaften.....	93
2.4.1	Auf ein Wort – die allmächtige Funktion llSetPrimitiveParams.....	93
2.4.2	Die Eigenschaft Physikalisch.....	93
2.4.3	Die Eigenschaft Phantom.....	96
2.4.4	Die Eigenschaft Temporär.....	96
2.4.5	Prim-Dimensionen.....	97
2.4.6	Die Position eines Prims innerhalb von Second Life.....	98
2.5	Formen.....	103
2.5.1	Pfadschnitt/Path Cut.....	106
2.5.2	Hohl und Hohlform/Hollow and Hole Shape.....	108
2.5.3	Verdrehung (Torsion)/Twist.....	110
2.5.4	Verjüngung/Taper.....	110
2.5.5	Scherung/Topshear.....	113
2.5.6	Vertiefung/Dimple.....	113
2.5.7	Lochgröße/Holesize.....	114
2.5.8	Profilschnitt/Profile Cut.....	116
2.5.9	Versatz/Skew.....	118
2.5.10	Umdrehungen/Revolutions.....	118
2.5.11	Spiralform/Radius Delta.....	119

2.6	Farben und Texturen von Objekten.....	121
2.6.1	Farben	121
2.6.2	Face, die Seiten eines Prims	122
2.6.3	Texturen	127
2.6.4	Transparente Prims/Objekte	130
2.7	Licht.....	131
2.8	Textliches.....	132
2.9	Die Funktion <code>lISetPrimitiveParams</code> im Überblick.....	133
2.9.1	Die Konstanten und Übergabewerte von <code>lISetPrimitiveParams</code>	133
2.9.2	Basisformen setzen mit <code>lISetPrimitiveParams</code>	135
2.9.3	Überblick über die Parameter der Formmanipulation	137
2.9.4	Konstanten für <code>Sculpted Prims</code>	138
2.9.5	Konstanten für <code>Bumpmapping</code>	138
2.9.6	Glanz-Konstanten.....	139
2.9.7	Prim-Material	139
3	Verknüpfte Objekte/Linked Sets	141
3.1	Allgemeines zu verknüpften Objekten.....	141
3.2	Anweisungen in verknüpften Objekten.....	142
3.2.1	Versenden von Nachrichten in <code>Linked Sets</code>	142
3.2.2	Empfangen von Nachrichten in <code>Linked Sets</code>	143
3.2.3	Eigenschaften abfragen und manipulieren in <code>Linked Sets</code>	144
3.2.4	Programmatisches Anlegen und Auflösen von verknüpften Objekten	150
4	Objektinventar	157
4.1	Allgemeines zum Objektinventar.....	157
4.2	WYGIWYG – eine Beispielanwendung	158
4.2.1	Das Basisskript.....	159
4.2.2	Einen neuen Gegenstand im Objektinventar ermitteln	162
4.2.3	Objektinventarrechte und Übergabe eines Gegenstands an einen Spieler	166
4.3	Auslesen von Notizen/Notecards	172
4.4	Skriptfunktionen	177
4.4.1	Auslesen von Landmarks	179
5	Kommunikation	181
5.1	Dialoge.....	181
5.2	Instant Messages	186
5.3	E-Mail.....	187
5.4	HTTP	190
5.5	XML-RPC	193
5.6	Beispielanwendung RSS-Reader	193
5.6.1	XyText	194
5.6.2	Das serverseitige Abrufen des RSS-Feeds	197
5.6.3	Aufruf des PHP-Skriptes aus <code>LindenScript</code> heraus.....	198
5.6.4	Der RSS-Reader als Ganzes	199
5.7	Finaler Überblick zur Kommunikation in <code>LindenScript</code>	202

6	Rotation, Animation und Translation eines nichtphysikalischen Objekts.....	205
6.1	Einführung.....	205
6.2	Rotation von Prims.....	206
6.2.1	Mathematisches.....	206
6.2.2	Umrechnung von Grad in Radiant und umgekehrt.....	207
6.2.3	Drehung eines Prims/Objektes.....	207
6.2.4	Wiederholte Drehung eines Prims/Objektes.....	208
6.2.5	Animierte Drehung mit IITargetOmega.....	211
6.2.6	Lokale Rotation in Linked Sets.....	213
6.3	Rotation und Animation von Texturen.....	215
6.3.1	Animation von Texturen durch Einzelbilder.....	217
6.3.2	Animierte Rotation von Texturen.....	218
6.3.3	Animation einer Textur durch Skalierung.....	218
7	Avatarfunktion.....	221
7.1	Allgemeine Avatarfunktionen.....	222
7.2	Sitzen.....	226
7.3	Attachments.....	230
7.4	Animation von Avataren.....	234
7.5	Steuerung von Avataren/Controls.....	239
7.6	Money, Money, Money.....	243
7.7	Teleports.....	253
8	Media.....	257
8.1	Audio.....	257
8.1.1	In-world sound.....	257
8.1.2	Streaming Audio.....	264
8.2	Video.....	266
9	Partikel.....	271
9.1	Einführung.....	271
9.2	Partikelemissionen beenden.....	273
9.3	Der Emitter.....	273
9.4	Die Funktion IIParticleSystem.....	274
9.4.1	Schemata/Pattern.....	275
9.4.2	Den Ausstoß von Partikeln beeinflussen.....	277
9.4.3	Farbe von Partikeln.....	278
9.4.4	Transparenz.....	279
9.4.5	Größe von Partikeln.....	280
9.4.6	Das Aussehen von Partikeln mit Texturen bestimmen.....	280
9.4.7	Rotation des Partikelemitters.....	282
9.4.8	Partikel auf ein Ziel zubewegen.....	282
9.4.9	Überblick über alle Übergabewerte an IIParticleSystem.....	284
9.5	Partikelbeispiele.....	287
9.5.1	Bling Bling.....	288
9.5.2	Let it snow, let it snow, let it snow.....	289
9.5.3	Feuer!!!.....	290

10	Erweiterte Objektfunktionalität	293
10.1	Huds.....	293
10.1.1	Vorabinformationen	293
10.1.2	Ein Hud-gesteuerter Bildbetrachter.....	295
10.2	Sensoren.....	299
10.2.1	Allgemeines zu Sensoren	300
10.2.2	Beispielprogramm Avatar-Scanner-Attachment	302
10.3	Physikalische Objekte.....	306
10.3.1	Beschleunigung/acceleration.....	307
10.3.2	Drehmoment/torque	308
10.3.3	Geschwindigkeit/velocity.....	309
10.3.4	Kraft/force.....	309
10.3.5	Masse/mass	310
10.3.6	Impuls/impulse.....	311
10.3.7	Reibung/friction	312
10.3.8	Schwerkraft/gravity.....	312
10.3.9	Energie/energy	312
10.3.10	Weitere Funktionen für physikalische Objekte	313
10.4	Kollisionen.....	314
10.5	Abschließendes Beispielprogramm – eine Pistole	316
11	Fahrzeuge	325
11.1	Grundkonzepte.....	325
11.2	Autos.....	327
11.2.1	Mein erstes Auto	327
11.2.2	Auto 2.0 – jetzt mit Reifen	334
11.2.3	Türen, Motorhaube und Kofferraum	336
11.2.4	Du nicht.....	337
11.3	Wasserfahrzeuge.....	339
11.4	Ein fliegender Teppich.....	343
11.5	Die Parameter der vehicle-Funktion im Überblick	345
A	Glossar.....	351
B	Alle Events im Überblick	355
C	Alternative Editoren	357
C.1	Compiler/Debugger	357
C.2	Editoren	357
C.3	Syntax-Checker.....	358
Register	359



1 Einführung in LindenScript

1.1 Die Entwicklungsumgebung

Wenn Sie Skripte in LindenScript schreiben möchten, sind Sie auf den Second Life-internen Skript-Editor angewiesen, zumindest solange Sie die Skripte *in-world* speichern möchten.

Der Editor kümmert sich hierbei um verschiedene Belange:

- Er stellt Ihnen ein Basisskript zur Verfügung (das wir uns im nächsten Abschnitt ansehen werden).
- Er überprüft Ihren Code beim Speichern auf Fehler und zeigt Ihnen diese Fehler an.
- Er veranlasst beim Speichern das Überführen Ihres Codes in einen Bytecode. Diesen Vorgang nennt man Kompilieren¹.
- Er kann Skripte für Sie neu starten. Alle Werte werden auf ihren Ausgangswert gesetzt.
- Er kann Skripte vorübergehend deaktivieren, sodass sie nicht ausgeführt werden, aber dennoch vorhanden sind. Dies regelt die Checkbox *Running*.
- Er stellt Ihnen eine Eingabehilfe zur Verfügung, mit der Sie Funktionen² per Klick auf die Drop-down-Liste *Insert* in Ihren Quellcode einfügen können.
- Er ist direkt mit dem LSL-Portal³ verbunden. Um eine Hilfestellung angezeigt zu bekommen, markieren Sie dazu eine Funktion und wählen aus der Menüleiste des Editors den Menüeintrag *Help* und dann *LSL Wiki Help ...*
- Er zeigt Ihnen einen Tooltip an, wenn Sie sich eine gewisse Zeit mit dem Cursor über einer Funktion befinden.

¹ <http://de.wikipedia.org/wiki/Kompilieren>

² Mehr dazu in Abschnitt 1.10 ab S. 48

³ http://wiki.secondlife.com/wiki/LSL_Portal



Abbildung 1.1 Der Skript-Editor von Second Life

Sie rufen den Skript-Editor durch folgende Schritte auf:

- Klicken Sie auf dem PC mit einem Rechtsklick auf ein Objekt.
- Wählen Sie aus dem erscheinenden Tortenmenü die Option *Bearbeiten*⁴.
- Klicken Sie den Button *Mehr >>* in der rechten unteren Ecke des erscheinenden Fensters.
- Wählen Sie den Reiter *Inhalt*.
- Klicken Sie auf *Neues Skript*.
- Unter dem angezeigten Ordner *Contents* erscheint ein Skript-Symbol mit dem Namen *New Script*. Doppelklicken Sie dieses.

⁴ Edit

Nach einer Zeit wird Sie der Second Life-interne Skript-Editor wahrscheinlich nicht mehr zufriedenstellen. Es fehlen einfach zu viele Features, die man aus anderen, erwachsenen, externen Editoren kennt, wie z.B. automatische Code-Komplettierung, um schneller arbeiten zu können. Daher werden Sie höchstwahrscheinlich im Laufe Ihrer Arbeit mit LindenScript auf einen externen Editor umsteigen wollen. Verschiedene externe Editoren werden im Anhang des Buches vorgestellt. Jedoch werden Sie zum Speichern eines Skriptes in Second Life immer wieder auf den internen Editor zurückgreifen müssen.

1.2 Hello, Avatar!

Sobald Sie Ihr erstes Skript angelegt haben und der Skript-Editor sich öffnet, sehen Sie bereits den ersten Code direkt vor sich. Dies ist die LindenScript-Variante von *Hello World*, einer Art erster Code, den es für wahrscheinlich jede Programmiersprache gibt⁵. Sinn und Zweck dieses Codes ist es, simple Anweisungen zu zeigen, die es benötigt, um eine Bildschirmausgabe zu generieren. In LindenScript heißt dieses Programm *Hello, Avatar!*.

Listing 1.1 Hello, Avatar!

```
default
{
    state_entry()
    {
        llSay(0, "Hello, Avatar!");
    }

    touch_start(integer total_number)
    {
        llSay(0, "Touched.");
    }
}
```

Lassen Sie uns doch einmal kurz den Code aus **Listing 1.1** gemeinsam durchgehen. Keine Angst, wenn Sie jetzt noch nicht verstehen, was hier eigentlich geschieht. Dies wird sich alles im Verlauf des Buches klären.

Die Anweisung der ersten Zeile, `default`, definiert einen `state`, im Deutschen: Zustand. Jedes Skript benötigt einen `default state`, damit es kompiliert werden kann.

Die geschweiften Klammern `{ }` dienen dazu, zusammengehörige Anweisungen zu identifizieren. Eine öffnende Klammer leitet einen Anweisungsblock ein, eine schließende Klammer beendet diesen wieder.

Der `state default` beinhaltet daher zwei weitere Anweisungen: den Event `state_entry` und den Event `touch_start`. Die beiden Ereignisse, so die deutsche Übersetzung von Event, sind wiederum Anweisungen, wie Sie anhand der geschweiften Klammern sehen können. Jeder `state` braucht mindestens einen Event, um ausführbar zu sein.

⁵ http://de.wikipedia.org/wiki/Liste_von_Hallo-Welt-Programmen

Sowohl in `state_entry` als auch im `touch_start`-Event wird eine einzelne Anweisung ausgeführt. Dies ist eine Funktion. Funktionen sind Zusammenfassungen von Anweisungsblöcken. Sie können sie selbst schreiben oder auf bereits bereitgestellte Funktionen zurückgreifen. Das Schöne an Funktionen ist, dass Sie nicht unbedingt wissen müssen, wie etwas in einer Funktion verarbeitet wird, aber Sie sich darauf verlassen können, was Sie von einer Funktion erwarten können.

Im Falle von *Hello, Avatar!* wird die Funktion

```
llSay(integer channel, string text)
```

aufgerufen. Diese Funktion veranlasst eine Bildschirmausgabe. Durch die Übergabe von Werten in den runden Klammern, sie werden Parameter genannt, weiß die Funktion, was sie tun soll. Der Funktion wurde der Wert `0` für den Parameter `channel` übergeben und die Zeichenkette `"Hello, Avatar!"` für den Parameter `text`. Der Parameter beschreibt dabei den Kanal, auf dem kommuniziert werden soll. Der Kanal `0` steht hierbei für den Haupt-Chat.

Die Funktion `llSay` verhält sich hierbei genauso, als würden Sie selbst etwas in den Haupt-Chat eingeben, nur dass als Absender nicht Ihr Name aufgeführt wird, sondern der Name des Objekts. Die Ausgabe durch `llSay` ist in einem Bereich von 20 Metern zu hören. Bei einer Übergabe von mehr als 1023 Zeichen wird die Ausgabe abgeschnitten.

Es gibt noch weitere Funktionen, mit denen Sie eine Bildschirmausgabe im Haupt-Chat erreichen können. Die Funktion

```
llWhisper(integer channel, string text)
```

verbreitet Nachrichten in einem Radius von zehn Metern. Die Ausgabe der Funktion

```
llShout(integer channel, string text)
```

hingegen kann noch in 100 Metern Entfernung wahrgenommen werden. Wenn Sie eine Ausgabe erzielen möchten, die nur Sie als Besitzer eines Objekts hören können, wählen Sie die Funktion

```
llOwnerSay(string message)
```

Dieser Funktion müssen Sie keinen Kanal angeben, da die Ausgabe ohnehin nur für Sie bestimmt ist und somit von anderen auch nicht wahrgenommen werden kann. `llOwnerSay` sollte im Großteil aller Fälle eingesetzt werden, wenn es Ihnen darum geht, Fehler in einem Skript aufzudecken bzw. Informationsmeldungen ausschließlich an Sie alleine abzusetzen. Dies gehört einfach zum guten Ton in Second Life. Damit stören Sie keine anderen Spieler in der Umgebung.

Dies war ein erster Überblick. Alle hier vorgestellten Konzepte von LindenScript werden im weiteren Verlauf dieses Kapitels noch genauer erläutert werden. Also dann, Ärmel hochkrepeln, jetzt geht es los!

1.3 Variablen

Eigentlich ist Programmieren gar nicht so schwer. Im Endeffekt geht es darum, ein paar einfache Basiskonzepte zu beherrschen, um diese dann kreativ anzuwenden. Eines dieser Konzepte ist der Einsatz von Variablen. Diese dienen dazu, Werte temporär im Speicher zu halten, damit Sie in der laufenden Verarbeitung des Skriptes darauf zurückgreifen können.

Es gibt viele Metaphern zur Erklärung von Variablen. Eine jedoch, die mir hängen geblieben ist, stammt von Colin Moog⁶. Er verglich hierbei Variablen mit Bankkonten. So wie man auf einem Bankkonto Geld einzahlen kann und dieses dann von der Bank für uns verwaltet wird, zahlen wir quasi Werte in unsere Variablen ein. Dabei kann es wie leider auch im richtigen Leben vorkommen, dass eine Variable einmal einen negativen Wert beinhaltet.

LindenScript kennt unterschiedliche Typen von Variablen. Um bei der Bankkonten-Metapher zu bleiben: Wir können nicht ohne weiteren Aufwand US-Dollar auf unser deutsches Konto einzahlen. Entweder brauchen wir dafür einen anderen Typ Bankkonto, oder wir bitten die Bank, eine Umrechnung für uns zu erledigen. Dies nennt man im Programmierer-Jargon *Casting*. Hierzu jedoch an geeigneter Stelle mehr.

LindenScript ist also eine typisierte Sprache, genauer gesagt eine stark typisierte Skriptsprache, und befindet sich mit seiner Typisierung in guter Gesellschaft zu anderen Programmiersprachen wie Java, C/C++ oder Python, um hier nur einige zu nennen.

Im weiteren Verlauf dieses Kapitels wird noch näher auf die unterschiedlichen Typen eingegangen werden, doch möchte ich Ihnen zunächst einmal einen Überblick über die verschiedenen in LindenScript implementierten Datentypen geben.

- String – Dient der Erfassung von Zeichenketten
- Integer – Zum Speichern ganzer Zahlen
- Float – Für Dezimalwerte
- Key – Ein einzigartiger Schlüssel, der in Second Life zur Identifizierung von Objekten, Avataren oder sonstigem verwendet wird
- Vector – Speichert drei `float`-Werte
- Rotation – Repräsentiert die Richtungsangabe eines Objektes in Second Life
- Quaternion – Identisch mit Rotation
- List – Eine Liste kann 0 oder mehrere Elemente anderer Datentypen speichern.

Das ist zunächst schon einmal einiges, was es zu erfassen gilt. Aber keine Sorge, ich werde im weiteren Verlauf genauer auf die einzelnen Datentypen eingehen.

Doch nun genug der Vorworte. Mit Sicherheit brennen Sie schon darauf, Ihre ersten Variablen anzulegen, um dann eilenden Schrittes Ihre ersten eigenen Skripte zu schreiben. Also sehen wir uns zunächst einmal an, wie Variablen deklariert werden.

⁶ aus seinem Buch ActionScript für Flash MX – Das Handbuch



Register

A

acceleration 307
agent 221, 225, 314, 351
AGENT_ALWAYS_RUN 225
AGENT_ATTACHMENTS 225
AGENT_AWAY 225
AGENT_BUSY 225
AGENT_CROUCHING 225
AGENT_FLYING 225
AGENT_IN_AIR 225
AGENT_MOUSELOOK 225
AGENT_ON_OBJECT 225
AGENT_SCRIPTED 225
AGENT_SITTING 225
AGENT_TYPING 225
AGENT_WALKING 225
ALL_SIDES 83, 122
Alternativer Avatar 160
Angular Deflection 331
Animation von Avataren 234
Ankerpunkt für Huds 293
Anlage Server 127
Anweisung 3, 39
Arithmetischer Operator 27
Array 19, 67
asset server 351
Asset Server 127
Asset-UUID kopieren 127
Assoziativgesetz 14
at_rot_target 355
at_target 313, 355
attach 230
Attachment 81, 221, 230, 239, 254, 271, 293, 302, 304, 320
Audio 257
Avatar 141, 142, 145, 183, 221, 306, 351
Avatar-Scanner 302

B

Banking 326, 341
Befehl 39
Beschleunigung 307

Bewegung 309
Bildbetrachter 295
Binärer Operator 33
Binärsystem 33
Bit 10
Bitfield 33, 91, 351
Bitweiser Operator 34
Bling 288
Block 39
boolean 30
Boolean 351
Boolescher Wert 31, 40, 351
Bounding Box 223
break 46
Bumpmap 352
Bumpmapping 138, 139
buoyancy 232
ByronStar 357
Bytecode 1

C

camelCase 7
Case sensitivity 7
Casting 5, 22, 25
changed 227, 233, 239, 355
CHANGED_ALLOW_DROP 162
CHANGED_ALLOWED_DROP 160
CHANGED_INVENTORY 162
Chat 4, 76
Chat-Kanal 77
Child Prim 352
Client 352
collision 309, 314, 355
collision_end 315, 316, 322, 355
collision_start 314, 316, 322, 355
Comma Separated Values 59
control 240, 332, 342, 345, 355
CONTROL_BACK 239
CONTROL_DOWN 239
CONTROL_FWD 239
CONTROL_LBUTTON 239

CONTROL_LEFT 239
CONTROL_ML_BUTTON 239
CONTROL_RIGHT 239
CONTROL_ROT_LEFT 239
CONTROL_ROT_RIGHT 239
CONTROL_UP 239
Controls 239

D

dangling-else-Problem 42
DATA_BORN 224
DATA_NAME 224
DATA_ONLINE 224
DATA_PAYINFO 224
DATA_RATING 224
dataserver 174, 175, 179, 223, 250, 355
Datentypen 5
DEBUG_CHANNEL 38
Debug-Fenster 128
Debug-Menü 126
default 3
DEG_TO_RAD 38, 207, 212
Dekrementoperator 28
Dezimalsystem 33
Dialogboxen 181
Dialoge 181
Dimple 113, 137
do 44
do-while 40, 44
Drehmoment 308, 309, 311
DRY 48

E

Editier-Modus 17
EHICLE_HOVER_TIMESCALE 341
else if 43
email 188, 189, 356
Email 203
Emitter 273
Energie 312
energy 312
EOF 38, 174
Ereignis 3, 75
Escape Code 12, 56
Escape Codes 352
Euler'sche Winkel 208
Event 3, 75
Explizite Typumwandlung 21

F

Face 122
Fahrzeuge 325
FALSE 30, 38
Farbe 121
Festkommazahl 11
Flag 34, 352

Float 5, 11
for 40, 44
force 309
friction 312
Full Bright 131
Funktion 48

G

Ganz hell 131
Ganzzahl 9
Geschwindigkeit 309
Glanz 139
Gleitkommazahl 11
globale Rotation 209, 337
globale Variable 7
GOTO 47
Gott-Modus 92
Grad 206
gravity 312
Grey Goo Fence 166
Grid 352
Gültigkeitsbereich 6, 47

H

Handle 76, 79
Havok 306
Heads-up-Display siehe Hud 293
Hello World 3
Hello, Avatar! 3
Hexadezimalzahl 10, 19
Hohl 108, 137
Hohlform 108, 137
Hole Shape 108
Holeshape 137
Holesize 114, 137
Hollow 108, 137
HTTP 203
HTTP_BODY_MAX_LENGTH 191
HTTP_BODY_TRUNCATED 193
HTTP_METHOD 191
HTTP_MIMETYPE 191
http_response 190, 193, 198, 201, 356
HTTP_VERIFY_CERT 191
HTTP-Request 190
Hud 230, 259, 271, 293

I

if 40
if-else 40, 41
Implizite Typumwandlung 25
Impuls 311
impulse 311
Inkrementoperator 28
Instant Messages 186
Integer 5, 9
INVENTORY_ALL 158

- INVENTORY_ANIMATION 158
 INVENTORY_BODYPART 158
 INVENTORY_CLOTHING 158, 161
 INVENTORY_GESTURE 158
 INVENTORY_NONE 158
 INVENTORY_NOTECARD 158
 INVENTORY_OBJECT 158
 INVENTORY_SCRIPT 158
 INVENTORY_SOUND 158
 INVENTORY_TEXTURE 158
 INVENTRORY_LANDMARK 158
 in-world sound 257
 LIST_STAT_MEAN 66
 Ivory Tower Library of Prims 120
- J**
- jump 40, 46
- K**
- Kameraposition 229
 Kanal 4
 Key 5, 19
 Kollision 139
 Kollisionserkennung 147
 Kommando 39
 Kommentar 10
 Kommunikation 181
 Kommunikationskanal 77
 Kommutativgesetz 14
 Konstante 30, 38
 Kontrolle 239
 Kontrollstrukturen 40
 Kraft 309
 Kreuzprodukt 15, 16, 17, 27
- L**
- Label 46
 Lag 76, 77, 79, 85, 160, 211, 271, 322, 335, 352
 land_collision 315, 356
 land_collision_end 315, 356
 land_collision_start 315, 356
 LibrarySenWarp 256
 LibraryWarpPos 255
 Licht 131
 Lichtquellen siehe Licht 131
 Linden Lab 352
 Linden Library 52
 Lindogramm 310, 311
 Linear Deflection 331
 Linebreak 132
 LINK_ALL_CHILDREN 143
 LINK_ALL_OTHERS 143
 link_message 143, 149, 164, 356
 LINK_ROOT 143
 LINK_SET 143
 LINK_THIS 143
 Linked Message 335, 337
 Linked Objects siehe Linked Sets 141
 Linked Set 88, 195, 199, 312, 315, 316, 325, 337, 352
 Linked Sets 97, 102, 141
 LinkedSet 306
 List 5, 19
 LIST_STAT_GEOMETRIC_MEAN 66
 LIST_STAT_MAX 66
 LIST_STAT_MEDIAN 66
 LIST_STAT_MIN 66
 LIST_STAT_NUM_COUNT 66
 LIST_STAT_RANGE 66
 LIST_STAT_STD_DEV 66
 LIST_STAT_SUM 66
 LIST_STAT_SUM_SQUARES 66
 listen 76, 78, 183, 356
 Listenvergleich 30
 llAdjustSoundVolume 258
 llAllowInventoryDrop 159, 161, 165, 169, 180
 llApplyImpulse 307, 310, 311
 llApplyRotationalImpulse 307, 308, 311
 llAttachToAvatar 230, 293
 llAvatarOnSitTarget 227
 llBreakAllLinks 155
 llBreakLink 155
 llCollisionFilter 316
 llCollisionSound 260, 315
 llCollisionSprite 315
 llCreateLink 150
 llCSV2List 59, 252
 llDeleteSubList 59, 69
 llDeleteSubString 54
 llDetachFromAvatar 233
 llDetectedGrab 302
 llDetectedGroup 302
 llDetectedKey 302
 llDetectedLinkNumber 146, 147, 302
 llDetectedName 302, 305
 llDetectedOwner 302
 llDetectedPos 302, 305
 llDetectedRot 302
 llDetectedType 302
 llDetectedVel 302, 307, 309
 llDialog 181, 203
 llDie 161, 233
 llDumpList2String 20, 51, 58
 llEmail 187, 251
 llEscapeURL 56, 191
 llEuler2Rot 207
 llFloor 78
 llFrand 78, 171
 llGetAccel 307
 llGetAgentInfo 225

IIGetAgentSize 145, 223
IIGetAlpha 130
IIGetAnimation 234
IIGetAnimationList 234
IIGetAttached 233
IIGetCenterOfMass 307, 311
IIGetColor 122
IIGetEnergy 307, 312
IIGetForce 307, 310
IIGetFreeMemory 75
IIGetInventoryCreator 180
IIGetInventoryName 161
IIGetInventoryNumber 160
IIGetInventoryPermMask 170, 180
IIGetInventoryType 161, 164
IIGetKey 90
IIGetLinkKey 145
IIGetLinkName 145
IIGetLinkNumber 145, 233
IIGetListEntryType 63
IIGetListLength 41, 51, 56
IIGetLocalPos 149, 294
IIGetLocalRot 149, 337
IIGetMass 307, 311
IIGetNextEmail 188, 356
IIGetNotecardLine 174, 175, 180, 355
IIGetNumberOfNotecardLines 174, 180, 355
IIGetNumberOfPrims 144
IIGetNumberOfSides 126
IIGetObjectDesc 90
IIGetObjectMass 307, 311
IIGetObjectName 90
IIGetObjectPermMask 91, 151, 170
IIGetOwner 150
IIGetPermissions 151
IIGetPos 98, 255, 305
IIGetPrimitiveParams 93, 135
IIGetRegionCorner 180
IIGetRootPosition 149
IIGetRootRotation 149
IIGetRot 209, 337
IIGetScale 97
IIGetStartParameter 153
IIGetStringLength 68
IIGetSubString 54, 58
IIGetTorque 307, 308
IIGetVel 307, 309, 332
IIGiveInventory 165, 172, 243
IIGiveInventoryList 165, 180
IIGiveMoney 244, 250, 251
IIHTTPRequest 186, 190, 192, 198, 356
IIInsertString 52
IIInstantMessage 203
IIKey2Name 77
IIList2CSV 59
IIList2Float 62
IIList2Integer 62
IIList2Key 62
IIList2List 58, 59, 68
IIList2ListStrided 68, 69
IIList2Rot 62
IIList2String 51, 62
IIList2Vector 62
IIListen 76, 77, 78, 181, 183, 356
IIListenControl 79
IIListenRemove 76
IIListInsertList 60
IIListLength 160
IIListListFindList 56
IIListRandomize 65, 67
IIListReplaceList 59
IIListSort 64, 67
IIListStatistics 66
IILoadURL 186
IILoopSound 259
IILoopSoundMaster 261
IILoopSoundSlave 261
IIMapDestination 254
IIMessageLinked 142, 149, 164, 170, 181, 195,
203, 251, 305, 356
IIMoveToTarget 313
IIOffsetTexture 129
IIOwnerSay 4, 51, 181, 203
IIParcelMediaCommandList 266, 268
IIParcelMediaQuery 267
IIParseString2List 60, 61
IIParseStringKeepNulls 61
IIParticleSystem 273, 274
IIPassCollisions 147, 315
IIPassTouches 146, 315
IIPlaySound 258
IIPreloadSound 259
IIPushObject 227, 307, 314
IIRegionSay 202
IIReleaseControls 242
IIRemoteLoadScriptPin 177, 180
IIRemoveInventory 162, 164, 180
IIRemoveVehicleFlags 347
IIRequestAgentData 223, 355
IIRequestInventoryData 179, 355
IIRequestPermissions 150, 230, 239, 245, 356
IIRequestSimulatorData 355
IIResetScript 81
IIRezObject 153, 166, 320, 356
IIRot2Fwd 320
IIRot2Left 320
IIRot2Up 320
IIRotTarget 355, 356
IISay 4, 181, 202
IIScaleTexture 129
IISensor 300, 301, 356
IISensorRemove 302

- llSensorRepeat 300, 301, 304, 356
 - llSetAlpha 128, 130
 - llSetBuoyancy 232, 307, 312, 322, 341
 - llSetCameraAtOffset 229
 - llSetCameraEyeOffset 229
 - llSetColor 83, 121
 - llSetDamage 322
 - llSetForce 307, 310, 312
 - llSetForceAndTorque 307, 310
 - llSetLinkAlpha 148
 - llSetLinkColor 148
 - llSetLinkPrimitiveParams 148, 214
 - llSetLinkTexture 148
 - llSetLocalRot 149, 337
 - llSetObjectDesc 89
 - llSetObjectName 89
 - llSetObjectPermMask 92
 - llSetParcelMusicURL 264
 - llSetPayPrice 243
 - llSetPos 98, 101, 149, 294
 - llSetPrimitiveParams 93, 95, 96, 98, 101, 102, 104, 106, 109, 111, 112, 115, 119, 122, 128, 129, 131, 133, 135, 138, 139, 208, 218, 255, 306, 316
 - llSetRemoteScriptAccessPin 177
 - llSetRot 208, 211, 337
 - llSetScale 97
 - llSetScriptState 177
 - llSetSitText 132, 229
 - llSetSoundQueueing 262
 - llSetStatus 94, 95, 306
 - llSetText 132, 169, 244, 297, 299, 305
 - llSetTexture 127
 - llSetTextureAnim 215, 217
 - llSetTimerEvent 79, 356
 - llSetTorque 307, 308
 - llSetTouchText 132
 - llSetVehicleFlags 332
 - llSetVehicleFloatParam 330
 - llSetVehicleRotationParam 330
 - llSetVehicleType 329, 340, 345
 - llSetVehicleVectorParam 330, 333
 - llShout 4, 181, 202
 - llSitTarget 226, 227, 239, 254
 - llSleep 170, 254
 - llStartAnimation 237
 - llStopAnimation 237
 - llStopMoveToTarget 314
 - llStopSound 260
 - llStringLength 41, 52
 - llSubStringIndex 54
 - llTakeControls 240, 242, 355
 - llTarget 313, 355, 356
 - llTargetOmega 211, 212, 334
 - llTeleportAgent 253
 - llTeleportAgentHome 253
 - llToLower 52
 - llToUpper 52
 - llTriggerSound 259, 260
 - llTriggerSoundLimited 259, 260
 - llUnescapeURL 56, 192
 - llUnsit 227
 - llVecDist 180, 305
 - llVecMag 309, 332
 - llVolumeDetect 316, 355
 - llWhisper 4, 181, 202
 - Lochgröße 114, 119, 137
 - Logischer Operator 31
 - lokale Rotation 209, 336
 - Lost and Found-Ordner 353
 - Lost And Found-Ordner 99
 - LSL-Editor 357
 - lslint 358
- ## M
- MASK_BASE 92, 170
 - MASK_EVERYONE 92
 - MASK_GROUP 92
 - MASK_NEXT 91
 - MASK_OWNER 92
 - mass 310
 - Masse 96, 310
 - Master-Slave-Funktionen 260
 - Material 139, 326
 - Matrizen-Addition 14
 - Methode 48
 - Modulo 27
 - Mond 131
 - money 243, 251, 356
 - Motor 326
 - Mouselook 320
 - moving_end 356
 - moving_start 356
 - MP3 257
 - Multidimensionaler Array 21, 67
 - multi-threading 76
 - Musik 257
- ## N
- n 12, 352
 - Newton 309
 - no_sensor 356
 - nosensor 301, 305
 - not_at_rot_target 356
 - not_at_target 313, 356
 - Notecards 172
 - Notizen 172
 - NULL_KEY 38

O

object_rez 153, 356
 Objekt 353
 Objektinventar 157
 Ogg Vorbis 258
 on_rez 76, 80, 90, 153, 233, 304, 322, 356
 Operator 12, 26

P

Parameterliste 48
 PARCEL_MEDIA_COMMAND_AGENT 266
 PARCEL_MEDIA_COMMAND_AUTO_ALIGN 266
 PARCEL_MEDIA_COMMAND_LOOP 266
 PARCEL_MEDIA_COMMAND_PAUSE 267
 PARCEL_MEDIA_COMMAND_PLAY 267
 PARCEL_MEDIA_COMMAND_STOP 267
 PARCEL_MEDIA_COMMAND_TEXTURE 267
 PARCEL_MEDIA_COMMAND_TIME 267
 PARCEL_MEDIA_COMMAND_UNLOAD 267, 269
 PARCEL_MEDIA_COMMAND_URL 267, 269
 particle system 284
 Partikel 271
 Partikel-Ausstoß berechnen 277
 Partikel-Farbe 278
 Partikel-Größe 280
 Partikeln-Texturen 280
 Partikel-Schemata 275
 Partikelsystem 273
 Partikel-Transparenz 279
 Path Cut 106, 137
 PAY_DEFAULT 244
 PAY_HIDE 244
 PERM_ALL 92
 PERM_COPY 92
 PERM_MODIFY 92
 PERM_MOVE 92
 PERM_TRANSFER 92
 PERMISSION_ATTACH 230, 233, 320
 PERMISSION_CHANGE_LINKS 150, 151
 PERMISSION_DEBIT 245
 PERMISSION_TAKE_CONTROLS 239, 320
 PERMISSION_TRIGGER_ANIMATION 151, 320
 Pfadschnitt 106, 137, 311, 337
 Phantom 96, 130, 314, 316
 Physics Engine 306
 physikalische Eigenschaft 94
 Physikalische Objekte 95, 306
 Physisch siehe physikalische Eigenschaft 94
 PI 38, 207
 PI_BY_TWO 38, 207
 Pick Color 121
 Poseball 237
 Prim 353
 Prim siehe Primitive 87
 PRIM_BUMP_BARK 138
 PRIM_BUMP_BLOBS 138
 PRIM_BUMP_BRICKS 138
 PRIM_BUMP_BRIGHT 138
 PRIM_BUMP_CHECKER 138
 PRIM_BUMP_CONCRETE 138
 PRIM_BUMP_DARK 138
 PRIM_BUMP_DISKS 138
 PRIM_BUMP_GRAVEL 138
 PRIM_BUMP_LARGETILE 139
 PRIM_BUMP_NONE 138
 PRIM_BUMP_SHINY 133, 138
 PRIM_BUMP_SIDING 139
 PRIM_BUMP_STONE 138
 PRIM_BUMP_STUCCO 139
 PRIM_BUMP_SUCTION 139
 PRIM_BUMP_TILE 138
 PRIM_BUMP_WEAVE 139
 PRIM_BUMP_WOOD 138
 PRIM_COLOR 133
 PRIM_FLEXIBLE 133
 PRIM_FULLBRIGHT 131, 134
 PRIM_HOLE_CIRCLE 109, 137
 PRIM_HOLE_DEFAULT 109, 137
 PRIM_HOLE_SQUARE 109, 137
 PRIM_HOLE_TRIANGLE 109, 137
 PRIM_MATERIAL 134, 139, 326
 PRIM_MATERIAL_FLESH 140
 PRIM_MATERIAL_GLASS 139
 PRIM_MATERIAL_METAL 139
 PRIM_MATERIAL_PLASTIC 140
 PRIM_MATERIAL_RUBBER 140
 PRIM_MATERIAL_STONE 139
 PRIM_MATERIAL_WOOD 139
 PRIM_PHANTOM 134
 PRIM_PHYSICS 134, 306
 PRIM_POINT_LIGHT 131, 134
 PRIM_POSITION 134, 255
 PRIM_ROTATION 117, 134
 PRIM_SCULPT_TYPE_CYLINDER 138
 PRIM_SCULPT_TYPE_PLANE 138
 PRIM_SCULPT_TYPE_SPHERE 138
 PRIM_SCULPT_TYPE_TORUS 138
 PRIM_SHINY_HIGH 139
 PRIM_SHINY_LOW 139
 PRIM_SHINY_MEDIUM 139
 PRIM_SHINY_NONE 139
 PRIM_SIZE 98, 134
 PRIM_TEMP_ON_REZ 97, 134
 PRIM_TEXGEN 134
 PRIM_TEXTURE 135
 PRIM_TYPE 119, 134
 PRIM_TYPE_BOX 135
 PRIM_TYPE_CYLINDER 135
 PRIM_TYPE_PRISM 135

PRIM_TYPE_RING 136
 PRIM_TYPE_SCULPT 136, 138
 PRIM_TYPE_SPHERE 113, 135
 PRIM_TYPE_TORUS 136
 PRIM_TYPE_TUBE 136
 Prim-Inventar 87, 127, 153, 156, 281
 Primitive 87, 353
 Priorität 234
 Profile Cut 116, 137
 Profilschnitt 116, 137
 Prozedur 48
 PSY_SRC_TEXTURE 280
 PSYS_PART_BOUNCE_MASK 287
 PSYS_PART_EMISSIVE_MASK 278, 287
 PSYS_PART_END_ALPHA 279, 284
 PSYS_PART_END_COLOR 278, 284
 PSYS_PART_END_SCALE 280, 284
 PSYS_PART_FLAGS 278, 284
 PSYS_PART_FOLLOW_SRC_MASK 287
 PSYS_PART_FOLLOW_VELOCITY_MASK 287
 PSYS_PART_INTERP_COLOR_MASK 278, 280, 287
 PSYS_PART_INTERP_SCALE_MASK 280, 287
 PSYS_PART_MAX_AGE 277, 285
 PSYS_PART_START_ALPHA 279, 285
 PSYS_PART_START_COLOR 278, 285
 PSYS_PART_START_SCALE 280, 285
 PSYS_PART_TARGET_LINEAR_MASK 283, 287
 PSYS_PART_TARGET_POS_MASK 282, 287
 PSYS_PART_WIND_MASK 287
 PSYS_SRC_ACCEL 285
 PSYS_SRC_ANGLE_BEGIN 275, 285
 PSYS_SRC_ANGLE_END 275, 285
 PSYS_SRC_BURST_PART_COUNT 277, 285
 PSYS_SRC_BURST_RADIUS 285
 PSYS_SRC_BURST_RATE 277, 286
 PSYS_SRC_BURST_SPEED_MAX 275, 286
 PSYS_SRC_BURST_SPEED_MIN 275, 286
 PSYS_SRC_MAX_AGE 282, 286
 PSYS_SRC_OMEGA 282, 286
 PSYS_SRC_PATTERN 275, 286
 PSYS_SRC_PATTERN_ANGLE 275, 286
 PSYS_SRC_PATTERN_ANGLE_CONE 275, 286, 289
 PSYS_SRC_PATTERN_ANGLE_CONE_EMPTY 277, 287
 PSYS_SRC_PATTERN_DROP 275, 287
 PSYS_SRC_PATTERN_EXPLODE 275, 287
 PSYS_SRC_TARGET_KEY 282, 286
 PSYS_SRC_TEXTURE 286
 Punktnotation 13
 Punktoperator 37

Q

Quaternion 5, 18, 206, 208
 QuickTime 266

R

RAD_TO_DEG 38, 207
 Radiant 206
 Radius Delta 119, 137
 Radius Offset 137
 Rechte-Hand-Regel 17
 Rechtevergabe 160
 Refactoring 85
 Region siehe Sim 166
 Reibung 139, 312
 Relationaler Operator 29
 remote_data 356
 Repeats per Face 129
 return 40
 Revolutions 118, 137
 Rezzen 353
 RGB-Format 121
 Root Prim 353
 Root-Prim 88
 rotation 205, 206, 207
 Rotation 5, 18, 205, 311, 334
 RSS 193
 run_time_permissions 151, 230, 239, 320, 356

S

Schattierungen 138
 Scherung 113
 Schlüssel 19, 90
 Schlüsselwort 48
 Schwerkraft 310, 312
 Scope 47
 Sculpted Prim 103, 353
 Sculpties 353
 Sculpties siehe Sculpted Prim 103
 SculptyPaint 104
 Seite eines Prims 122
 sensor 301, 304, 309, 356
 SetLocalRot 213
 Sim 77, 98, 165, 179, 187, 193, 202, 221, 225, 233, 301, 311, 353
 Sit Here 132
 Sit-Teleport-Hack 254
 Sitzen 226
 Skalarprodukt 15
 Skew 118, 137
 Skript-Editor 1
 Sonne 131
 Sound 139, 257
 Spaltenmatrix 13
 Speed 309
 Spielerinventar 157

Spiralform 119, 137
sprintf 173
SQRT2 38
Stack-Heap Collision Error 75
start_param 322
state 3, 40, 82
state_entry 3, 75, 356
state_exit 82, 356
STATUS_BLOCK_GRAB 95
STATUS_CAST_SHADOWS 95
STATUS_DIE_AT_EDGE 95
STATUS_PHANTOM 95, 96
STATUS_PHYSICAL 334
STATUS_PHYSICS 94, 306
STATUS_RETURN_AT_EDGE 95
STATUS_ROTATE_X 94
STATUS_ROTATE_Y 94
STATUS_ROTATE_Z 94
STATUS_SANDBOX 95, 99
Streaming Audio 257, 264
Strided List 66
Strided Lists 64, 67
Strides 67
String 5, 11
Syntax Error 6

T

t 12, 352
Taper 110, 137
Teleport 90, 233, 253
Temporäre Objekte 97
temporäres Prim-Limit 97
Temporary siehe Temporäre Objekte 97
Textur 121, 127, 131, 266
The Particle Laboratory 287
timer 76, 79, 189, 356
Timer 99
Topshear 113, 137
torque 308
Torsion 110, 137
touch 146, 254, 309, 356
Touch 132
touch_end 146, 356
touch_start 3, 75, 146, 356
Transparenz 128, 130
TRUE 30, 38
Twist 110, 137
TWO_PI 38, 218
TYPE_FLOAT 63
TYPE_INTEGER 63
TYPE_INVALID 63
TYPE_KEY 63
TYPE_ROTATION 63
TYPE_STRING 63
TYPE_VECTOR 63
Typumwandlung 21

U

Umdrehungen 118, 137
Unärer Operator 37
Universal Unique Identifier siehe UUID 127
Unix-Timestamp 189
UTF-8 56
UUID 127, 353

V

Variable 5
Vector 5, 12
VEHICLE_ANGULAR_DEFLECTION_EFFICIENCY 330, 347
VEHICLE_ANGULAR_DEFLECTION_TIMESCALE 331, 347
VEHICLE_ANGULAR_FRICTION_TIMESCALE 332, 348
VEHICLE_ANGULAR_MOTOR_DECAY_TIMESCALE 330, 347
VEHICLE_ANGULAR_MOTOR_DIRECTION 333, 348
VEHICLE_ANGULAR_MOTOR_TIMESCALE 330, 347
VEHICLE_BANKING_EFFICIENCY 341, 347
VEHICLE_BANKING_MIX 341, 347
VEHICLE_BANKING_TIMESCALE 341, 347
VEHICLE_BUOYANCY 341, 347
VEHICLE_FLAG_CAMERA_DECOUPLED 346
VEHICLE_FLAG_HOVER_GLOBAL_HEIGHT 342, 346
VEHICLE_FLAG_HOVER_TERRAIN_ONLY 346
VEHICLE_FLAG_HOVER_UP_ONLY 342, 346
VEHICLE_FLAG_HOVER_WATER_ONLY 342, 346
VEHICLE_FLAG_LIMIT_MOTOR_UP 332, 333, 346
VEHICLE_FLAG_MOUSELOOK_BANK 346
VEHICLE_FLAG_MOUSELOOK_STEER 346
VEHICLE_FLAG_NO_DEFLECTION_UP 347
VEHICLE_HOVER_EFFICIENCY 340, 347
VEHICLE_HOVER_HEIGHT 340, 342, 347
VEHICLE_HOVER_TERRAIN_ONLY 342
VEHICLE_HOVER_TIMESCALE 348
VEHICLE_LINEAR_DEFLECTION_EFFICIENCY 330, 348
VEHICLE_LINEAR_DEFLECTION_TIMESCALE 331, 348
VEHICLE_LINEAR_FRICTION_TIMESCALE 331, 348
VEHICLE_LINEAR_MOTOR_DECAY_TIMESCALE 330, 348
VEHICLE_LINEAR_MOTOR_DIRECTION 333, 348
VEHICLE_LINEAR_MOTOR_OFFSET 348

VEHICLE_LINEAR_MOTOR_TIMESCALE
330, 348
VEHICLE_REFERENCE_FRAME 341, 349
VEHICLE_TYPE_AIRPLANE 345, 346
VEHICLE_TYPE_BALLOON 346
VEHICLE_TYPE_BOAT 340, 346
VEHICLE_TYPE_CAR 329, 346
VEHICLE_TYPE_NONE 346
VEHICLE_TYPE_SLED 346
VEHICLE_VERTICAL_ATTRACTION_EFFICIE
NCY 331, 348
VEHICLE_VERTICAL_ATTRACTION_TIMESCA
ALE 331, 348
Vektor 12
velocity 309
Vendor 243, 250
Verdrehung 110, 137, 311
Vergleichsoperator 29
Verjüngung 110, 137
Verknüpfte Objekte 141
Verknüpfung aufheben 88
Verknüpfung erstellen 88
Versatz 118, 137
Versicherung 137
Vertical Attractor 331, 341
Vertiefung 113, 137
Video 257, 266

W

warpPos 255
WAV 257
Wertzuweisung 8
while 40, 43
Wiederholungen pro Gesicht 129
Winkel 206
Wortbreite 10, 11

X

XMLRPC 190, 193, 203
XyText 133, 194

Y

PSYS_SRC_ANGLE_END 285

Z

Zeichenkette 11
Zeilenmatrix 13
Zeilennummerierung 6
ZERO_ROTATION 19, 38
ZERO_VECTOR 38
Zugriffsrechte 91
Zustand 3, 82
Zuweisungsoperator 26
Zweierkomplement 33