SpeedText 2

Funktionsreferenz

Copyright © Christian Klaussner

1. Funktionen

Function **TextInitialize**(buffer)

Initialisiert die SpeedText-Bibliothek. Diese Funktion muss vor jeder anderen Funktion dieser Bibliothek aufgerufen werden. Andernfalls schlagen alle anderen Funktionen fehl.

Parameterbufferindiziert den Standard-Rendering-Puffer. Normalerweise wird BackBuffer() als Standardpuffer verwendet. Sie können es nach der Initialisierung ändern, indem Sie TextSetBuffer aufrufen.

ReturnsTrueif Initialisierung war erfolgreich, ansonsten False.

Function **TextDeinitialize**()

Deinitialisiert die SpeedText-Bibliothek und löscht alle ihre Ressourcen (außer Schriftarten, die mit TextLoadFont erstellt wurden.Sie sollten sie mit TextFreeFont vor der Deinitialisierung löschen).

ReturnsTrueif Deinitialisierung war erfolgreich, ansonsten False.

Function **TextSetBuffer**(buffer)

Setzt den aktuellen Rendering-Puffer. Dieser Puffer ist das Ziel für TextDrawandTextDrawRect.

Function **TextGetBuffer**()

Gibt den aktuellen Rendering-Puffer zurück.

Function **TextFreeFont**(font)

Löscht die angegebene Schriftart. Nach Aufruf dieser Funktion können Sie das Schrifthandle nicht mehr verwenden.

Function **TextSetFont**(font)

Setzt die aktuelle Schriftart für Zeichnungs- und Metrikberechnungen wie TextStringWidthand TextFontWidth. Geben Sie Zeroforfont an, um die Standardschrift (Courier) zu verwenden.

Function **TextGetFont**()

Gibt die aktuell ausgewählte Schriftart zurück. Wenn keine Schriftart mit TextSetFont ausgewählt wurde, ist der Rückgabewert Zero.

Function **TextLoadFont**(fontname$, height, bold, italic, underline, quality, filename$)

Lädt eine Schrift mit den angegebenen Eigenschaften. Parameterfontname$gibt den Namen der Schriftart an, z.B. "Times New Roman". Wenn die Schriftart nicht auf dem System installiert ist, sondern als TTF- oder OTF-Datei verfügbar ist, können Sie den Dateinamen für parameterfilename $angeben. Beachten Sie, dass der wahre Name der Schriftart noch angegeben werden muss.

Parameterheight bestimmt die Schrifthöhe in Pixeln. Parameter bold,italicandunderline können auf TrueorFalse gesetzt werden.

Parameter quality kann einer der folgenden Werte sein:

TEXT_DEFAULT	Verwenden Sie die vom Betriebssystem verwendeten Qualitätseinstellungen.
TEXT_NONANTIALIASED	Verwenden Sie keine Anti-Aliasing.
TEXT_ANTIALIASED	Verwenden Sie Anti-Aliasing.
TEXT_CLEARTYPE	ClearType Anti-Aliasing verwenden.

Gibt das neue Schrifthandle zurück, wenn das Laden erfolgreich war, ansonsten Zero.

Function **TextDraw**(x, y, text$, ax, ay, encoding)

Zeichnet Text in den aktuell ausgewählten Puffer an Position (x,y). Parameter xanday geben die Ausrichtung für die jeweiligen Achsen an.

Werte für die horizontale Ausrichtung:

TEXT_LEFT	Der Bezugspunkt befindet sich am linken Rand des Textes.
TEXT_CENTER	Der Text ist horizontal zentriert am Bezugspunkt.
TEXT_RIGHT	Der Bezugspunkt befindet sich am rechten Rand des Textes.

Werte für die vertikale Ausrichtung:

TEXT_TOP	Der Bezugspunkt befindet sich am oberen Rand des Textes.
TEXT_MIDDLE	Der Text ist vertikal zentriert am Bezugspunkt.
TEXT_BOTTOM	Der Bezugspunkt befindet sich am unteren Rand des Textes.

Gültige Codierungstypen finden Sie unter KapitelText-Codierung.

Function **TextDrawRect**(x, y, width, height, text$, ax, format, encoding)

Zeichnet Text mit einem Rechteck und Formatierungsoptionen in den aktuell ausgewählten Puffer. Die Formatierung basiert auf dem Rechteck, das von widethandheight angegeben wird. Wenn Sie die Größe des Rechtecks nicht kennen, geben Sie -1 für beides an.
Parameter können entweder TEXT_LEFT, TEXT_CENTER oder TEXT_RIGHT sein. Siehe TextDraw für

Details. Der Parameter kann einer der folgenden Werte sein:

TEXT_WORDWRAP	Zeilen werden gebrochen, wenn ein Wort nicht in das Rechteck passt.
TEXT_DONTCLIP	Text, der nicht in das Rechteck passt, wird nicht beschnitten.

Gültige Codierungstypen finden Sie unter KapitelText-Codierung.

Function **TextLockBuffer**()

Sperrt den aktuell ausgewählten Puffer für eine schnellere Textwiedergabe. Ein Puffer kann nur für die Textwiedergabe verwendet werden, während er gesperrt ist.

Function **TextUnlockBuffer**()

Entsperrt den aktuell ausgewählten Puffer.

Function **TextFontWidth**()

Gibt die Breite des breitesten Zeichens in der aktuell ausgewählten Schriftart zurück.

Function **TextFontHeight**()

Gibt die Höhe des höchsten Zeichens in der aktuell ausgewählten Schriftart zurück.

Function **TextFontAscent**()

Gibt den Aufstieg der aktuell ausgewählten Schriftart zurück. Die Summe von Auf- und Abstieg einer Schrift ist gleich ihrer Höhe.

Function **TextFontDescent**()

Gibt den Abstieg der aktuell ausgewählten Schriftart zurück. Die Summe von Auf- und Abstieg einer Schrift ist gleich ihrer Höhe.

Function **TextStringWidth**(text$, encoding)

Gibt die Breite des Stringtexts zurück. Die aktuell ausgewählte Schriftart wird zur Messung verwendet.

Gültige Codierungstypen finden Sie unter KapitelText-Codierung.

Function **TextStringHeight**(text$, encoding)

Gibt die Höhe des Stringtexts zurück. Die aktuell ausgewählte Schriftart wird zur Messung verwendet.

Gültige Codierungstypen finden Sie unter KapitelText-Codierung.

Function **TextSetColor**(red, green, blue)

Legt die Farbe für die Textwiedergabe im RGB-Format fest.

Function **TextSetBackground**(red, green, blue)

Legt die Hintergrundfarbe für die Textwiedergabe im RGB-Format fest. Geben Sie -1 für alle Farben an, um den Hintergrund transparent zu machen.

Function **TextColorRed**()

Gibt die rote Komponente der aktuell ausgewählten Farbe zurück.

Function **TextColorGreen**()

Gibt die grüne Komponente der aktuell ausgewählten Farbe zurück.

Function **TextColorBlue**()

Gibt die blaue Komponente der aktuell ausgewählten Farbe zurück.

Function **TextBackgroundRed**()

Gibt die rote Komponente der aktuell ausgewählten Hintergrundfarbe zurück.

Function **TextBackgroundGreen**()

Gibt die grüne Komponente der aktuell ausgewählten Hintergrundfarbe zurück.

Function **TextBackgroundBlue**()

Gibt die blaue Komponente der aktuell ausgewählten Hintergrundfarbe zurück.

2. Textkodierung

Text kann mit ANSI oder Unicode (UTF-8) kodiert werden. Die Standard Blitz3D IDE verwendet ANSI Strings, die für die meisten westlichen Sprachen wie Englisch oder Deutsch geeignet sind. Wenn Sie asiatischen oder arabischen Text benötigen, müssen Sie ihn im UTF-8 Format mit einem geeigneten Editor wie Notepad speichern. Alle SpeedText-Funktionen, die Textzeichenfolgen als Parameter akzeptieren, haben eine

3. Leistungsfragen

Bei der Verwendung von SpeedText unter Windows Vista liegt ein Leistungsproblem vor. Die Zeichenleistung von Standard-Blitz3D-Texten ist aufgrund der Grafikarchitektur von Windows Vista in den meisten Fällen schneller als SpeedText. Dieses Problem tritt unter Windows XP und Windows 7 RC nicht auf.