GM RegEdit DLL

Version 1.00

Der Game Maker bietet zwar selber Funktionen, die mit der Windows Registry umgehen können, jedoch in eingeschränktem Funktionsumfang. Gerade wenn man bestimmte Datentypen der Registry schreiben oder lesen will oder Schlüssel und Eintrage löschen will, stößt der Game Maker an seine Grenzen. Die *RegEdit DLL* bietet diese Möglichkeiten.

Funktionen

Falls im folgenden eine Funktion als Argument das Verzeichnis in der Registry verlangt, muss dieses relativ zum Hauptverzeichnis (z.B. `HKEY_LOCAL_MACHINE`) ohne führendes Trennzeichen angegeben werden und als Trennzeichen „`\\`", also zwei Backslash, benutzt werden. Hier ein Beispiel für eine gültige Verzeichnisangabe:

```
SOFTWARE\\Microsoft\\Windows Media
```

`reg_init()`

Initialisiert alle Funktionen der DLL. Muss aufgerufen werden vor der Nutzung aller weiteren Funktionen.

`reg_set_root(`*`key`*`)`

Setzt das Hauptverzeichnis der Registry. Alle Verzeichnisangaben in den folgenden Funktionen beziehen sich dann auf dieses Verzeichnis. Dabei muss `key` einen Zahlenwert mit folgender Bedeutung besitzen.

```
0 = HKEY_CLASSES_ROOT
1 = HKEY_CURRENT_USER
2 = HKEY_LOCAL_MACHINE
3 = HKEY_USERS
4 = HKEY_CURRENT_CONFIG
```

`reg_get_type(`*`key`*`,`*`value`*`)`

Liefert den Typ eines Wertes in der Registry, wobei `key` für das Verzeichnis und `value` für den Wert steht. Die Funktion liefert einen Wert, der für folgende Typen steht, oder `-1` falls es sich um einen anderen Typ handelt.

```
0 = REG_SZ
1 = REG_EXPAND_SZ
2 = REG_MULTI_SZ
3 = REG_BINARY
4 = REG_DWORD
5 = REG_QWORD
```

`reg_exists_value(`*`key`*`,`*`value`*`)`

Gibt zurück, ob der Wert `value` im angegebenen Verzeichnis `key` existiert.

`reg_exists_key(`*`key`*`)`

Gibt zurück, ob das angegebenen Verzeichnis `key` existiert.

`reg_create_key(`*`key`*`)`

Legt ein neues Verzeichnis `key` an und gibt zurück, ob das Anlegen erfolgreich war. Beachte, dass du nicht mehrere Unterverzeichnisse auf einmal anlegen kannst, sondern dafür ein mehrmaliger Funktionsaufruf nötig ist.

`reg_delete_value(`*`key`*`,`*`value`*`)`

Löscht den Wert `value` im Verzeichnis `key` und gibt zurück, ob der Vorgang erfolgreich war.

`reg_delete_key(`*`key`*`)`

Löscht das Verzeichnis `key` und gibt zurück, ob es gelöscht werden konnte.

`reg_read_string(`*`key`*`,`*`value`*`[,`*`index`*`])`

Gibt den Wert `value` im Verzeichnis `key` zurück. Mit dieser Funktion lassen sich Werte vom Typ einer Zeichenfolge lesen. Das umfasst folgende Typen.

```
REG_SZ
REG_EXPAND_SZ
REG_MULTI_SZ
```

Der Typ `REG_MULTI_SZ` ist ein Feld von Zeichenfolgen. Mit der Funktion lässt sich nur eine dieser Zeichenfolgen abrufen. Dafür existiert der Parameter `index`, welcher die Position angibt. Dabei steht 0 für die erste Zeichenfolge. Falls `index` einen größeren Wert angibt, als Zeichenfolgen vorhanden sind, wird eine leere Zeichenkette zurückgegeben.

`reg_read_binary(`*`key`*`,`*`value`*`)`

Liest den Wert `value` im Verzeichnis `key` vom Typ `REG_BINARY` und gibt diesen in Hexadezimaldarstellung zurück. z.B. würde der Wert „012345“ zurückgegeben als „303132333435“.

`reg_read_int(`*`key`*`,`*`value`*`)`

Gibt den Wert `value` aus dem Verzeichnis `key` zurück. Dabei muss es sich um einen Zahlentyp handeln, also `REG_DWORD` oder `REG_QWORD`.

`reg_write_string(`*`key`*`,`*`value`*`,`*`data`*`,`*`type`*`)`

Schreibt die Zeichenkette `data` in den Wert `value` im Verzeichnis `key`. Dabei legt `type` den Typ der Zeichenkette fest. Dafür sollte einer der folgenden Zahlenwerte übergeben werden.

```
0 = REG_SZ
1 = REG_EXPAND_SZ
2 = REG_MULTI_SZ
```

Für den Typ `REG_MULTI_SZ` gibt es eine Besonderheit. Dieser Typ ist für ein Feld von Zeichenfolgen gedacht. Falls `value` noch nicht existiert oder nicht vom Typ `REG_MULTI_SZ` ist, wird der Wert einfach auf `data` gesetzt. Anderenfalls wird `data` an das Feld von Zeichenfolgen angehangen. Wird für `data` eine leere Zeichenfolge übergeben, wird diese nicht zum Feld hinzugefügt sondern `value` wird auf eine leere Zeichenfolge gesetzt.

Die Funktion gibt zurück, ob der Wert eingetragen werden konnte.

`reg_write_binary(`*`key`*`,`*`value`*`,`*`data`*`)`

Schreibt `data` als Typ `REG_BINARY` auf den Wert `value` im Verzeichnis `key`. Dabei muss `data` in Hexadezimaldarstellung übergeben werden. Also z.B. „48414C4C4F“ für die Zeichenfolge „HALLO “.

Es wird zurückgegeben, ob das Schreiben erfolgreich war.

`reg_write_int(`*`key`*`,`*`value`*`,`*`data`*`,`*`type`*`)`

Schreibt den Zahlenwert `data` in den Wert `value` im Verzeichnis `key`. Mit `type` wird der Typ festgelegt. Das muss einer der beiden folgenden beiden Werte sein.

```
4 = REG_DWORD  (32-Bit Wert)
5 = REG_QWORD  (64-Bit Wert)
```

Die Funktion gibt zurück, ob das Schreiben erfolgreich war.