Sound

COLLABORATORS			
	TITLE : Sound		
ACTION	*NAME*	*DATE*	*SIGNATURE*
WRITTEN BY		January 23, 2025	

REVISION HISTORY			
NUMBER	DATE	DESCRIPTION	NAME

Contents

Chapter 1

Sound

1.1 Sound V1.03

```
PureBasic - Sound V1.03

  Das Verwalten von Sounds wird niemals so einfach und
  schnell möglich sein wie mit dieser Library. Glauben
  Sie uns, wir nutzen die Amiga Hardware bis an ihre
  Grenzen, um systemkompatibles aber auch unmittelbares
  Abspielen von Sounds zu gewährleisten. Es werden
  einfach die 4 Stereo Audio-Kanäle für eine effiziente
  Ausgabe genutzt. Das Sound-Format ist IFF/8SVX, dem
  Standard auf dem Amiga.

Befehlsübersicht:

   AllocateSoundChannels
   ChangeSoundPeriod
   ChangeSoundVolume
   CopySound
   CreateSound
   DecodeSound
   FreeSound
   FreeSoundChannels
   GetSoundLength
   InitSound
   LoadSound
   PeekSoundData
   PlaySound
   PokeSoundData
   SaveSound
   SetSoundChannels
   SetSoundPeriod
   SetSoundVolume
   SoundFilter
   StopSound

Beispiel:

   Sound Demo
```

1.2 allocatesoundchannels

```
  SYNTAX
Result.w = AllocateSoundChannels(Channels.w)

  FUNCTION
Benutzen Sie diese Funktion, um einen oder mehrere Kanäle zu reservieren.

Channels
Diese Maske gibt an, welche Kanäle reserviert werden sollen.

1 = versucht, Kanal 0 zu reservieren
2 = versucht, Kanal 1 zu reservieren
4 = versucht, Kanal 2 zu reservieren
8 = versucht, Kanal 3 zu reservieren

Nach Addition der Werte:

 9 = versucht, die Kanäle 0 und 3 zu reservieren
15 = versucht, alle Kanäle zu reservieren

Result
Ist diese Maske TRUE, zeigt sie an, welche Kanäle reserviert wurden.
Andernfalls ergibt sie FALSE und keiner der gewünschten Kanäle konnte
reserviert werden.
```

1.3 changesoundperiod

```
  SYNTAX
Result.w = ChangeSoundPeriod(#Obj.w,Period.w)

  FUNCTION
Benutzen Sie diese Funktion, um die Periode während dem
Abspielen des Sounds zu verändern.

Die im Sound Objekt festgelegte Periode, welche beim
Start des Sounds benutzt wird, wird mit dieser Funktion
nicht verändert.

#Obj
Das Objekt, bei welchem die Periode verändert werden soll.

Period
Dies ist der neue Perioden-Wert.

Result
Diese Maske zeigt an, welches die von der Periodenänderung
betroffenen Kanäle sind.
```

1.4 changesoundvolume

```
  SYNTAX
Result.w = ChangeSoundVolume(#Obj.w,Volume.w)

  FUNCTION
Benutzen Sie diese Funktion, um die Lautstärke während dem
Abspielen des Sounds zu verändern.

Die im Sound Objekt festgelegte Lautstärke, welche beim
Start des Sounds benutzt wird, wird durch diese Funktion
nicht verändert.

#Obj
Das Objekt, bei welchem die Lautstärke verändert werden soll.

Volume
Dies ist der neue Lautstärken-Wert.

Result
Diese Maske zeigt an, welches die von der Lautstärkenänderung
betroffenen Kanäle sind.
```

1.5 copysound

```
  SYNTAX
Result.l = CopySound(#Sound.w,#Sound.w)

  FUNCTION
Diese Funktion erstellt aus einem bereits existierenden #Soundobjekt
eine komplette Kopie.

#Sound
Der zu kopierende Sound.

#Sound
Der neue Sound.

Result
Ist dieses TRUE, wurde die Kopie von #Sound erstellt, andernfalls
ist das Ergebnis FALSE.
```

1.6 createsound

```
  SYNTAX
Result.l = CreateSound(#Obj.w,Length.l)

  FUNCTION
Diese Funktion erstellt ein neues Sound Objekt und setzt
dessen gesamte Sound Daten auf NULL. Periode, Lautstärke
und Kanäle werden ebenfalls auf NULL gesetzt.

Um die Sound-Daten lesen und schreiben zu können, benutzen
```

```
Sie PeekSoundData() und PokeSoundData(). Periode, Lautstärke
und Kanäle müssen mit den entsprechenden Funktionen festgelegt
werden.

#Obj
Das zu erstellende Objekt.

Length
Die Länge der Sound-Daten.

Es ist nutzlos, hier irgendeinen Wert größer als 128 KB
anzugeben, da die Amiga Hardware dies nicht unterstützt.

Result
Ergibt dieses TRUE, konnte das Objekt erstellt werden,
andernfalls wird FALSE zurückgegeben.
```

1.7 decodesound

```
  SYNTAX
Result.w = DecodeSound(#Obj.w,Pointer.l)

  FUNCTION
Diese Funktion initialisiert ein Sound Objekt mit dem
innerhalb des Programms gespeicherten IFF Sound.

* Periode wird auf den - der IFF Datei entnommenen - Wert gesetzt.
* Lautstärke (Volume) wird auf 64 gesetzt.
* Kanäle werden auf 15 gesetzt.

Ist das Sound Objekt bereits initialisiert, muß es erst mit
dem Aufruf der Funktion FreeSound() freigegeben werden.
Andernfalls verbleibt es im Speicher, aber es gibt keine
Möglichkeit mehr, es abzuspielen.

#Obj
Das zu benutzende Objekt.

Pointer
Ein Zeiger (Pointer) auf den IFF Sound.

Result
Dies ergibt TRUE, wenn das Sound Objekt mit dem eingebundenen
Sound initialisiert werden konnte, andernfalls FALSE.
```

1.8 freesound

```
  SYNTAX
FreeSound(#Obj.w)

  STATEMENT
Dieser Befehl gibt ein Sound Objekt frei, welches mit
```

```
LoadSound() oder DecodeSound() initialisiert wurde.

Ist kein FastRam Speicher verfügbar, wird schließlich
ChipRam Speicher benutzt. Ein mit DecodeSound()
initialisiertes Sound Objekt bleibt gültig, bis es mit
LoadSound() oder mit DecodeSound() (zusammen mit
einem eingebundenen IFF Sound) erneut initialisiert wird.
Dies bedeutet, dass ein solches Objekt niemals freigegeben
werden kann.

#Obj
Das freizugebende Objekt.
```

1.9 freesoundchannels

```
  SYNTAX
Result.w = FreeSoundChannels(Channels.w)

  FUNCTION
Diese Funktion gibt einen oder mehrere Kanäle frei.

Channels
Diese Maske gibt an, welche Kanäle freigegeben werden sollen.

1 = gibt Kanal 0 frei
2 = gibt Kanal 1 frei
4 = gibt Kanal 2 frei
8 = gibt Kanal 3 frei

Nach Addition der Werte:

 9 = gibt beide Kanäle 0 und 3 frei
15 = gibt alle Kanäle frei

Result
Ist diese Maske TRUE zeigt sie an, welche Kanäle freigegeben
wurden. Andernfalls ist sie FALSE und dies bedeutet, sie wurden
nicht freigegeben, weil sie niemals reserviert wurden.
```

1.10 getsoundlength

```
  SYNTAX
Result.l = GetSoundLength(#Obj.w)

  FUNCTION
Diese Funktion ermittelt die Länge eines Sound Objektes.

Benutzen Sie diesen Wert für entsprechende Berechnungen,
damit PeekSoundData() und PokeSoundData() nicht außerhalb
der aktuellen Sound Daten lesen oder schreiben.

#Obj
```

```
Das zu benutzende Objekt.

Result
Die Länge der Sound Daten.
```

1.11 initsound

```
  SYNTAX
Result.w = InitSound(Objects.l)

  FUNCTION
Diese Funktion ist die Initialisierungsroutine, um die für
die Sounds benötigte Programmumgebung einzurichten. Sie muß
vor allen anderen Funktionen aufgrufen werden. Sie kann nur
einmal aufgerufen werden.

Objects
Dieser Wert gibt an, wieviele Sound Objekte gewünscht werden.

Result
Ergibt dieses TRUE, war der Aufruf erfolgreich. Andernfalls
wird FALSE zurückgegeben und keine anderen Funktionen dieser
Library können aufgerufen werden
```

1.12 loadsound

```
  SYNTAX
Result.b = LoadSound(#Obj.w,FileName$)

  FUNCTION
Benutzen Sie diese Funktion, um ein Sound Objekt mittels einem
auf Disk gespeicherten IFF Sound zu initialisieren.

* Periode wird auf den - der IFF Datei entnommenen - Wert gesetzt.
* Lautstärke (Volume) wird auf 64 gesetzt.
* Kanäle werden auf 15 gesetzt.

Ist das Sound Objekt bereits initialisiert, muß es erst mit
dem Aufruf der Funktion FreeSound() freigegeben werden.
Andernfalls verbleibt es im Speicher, aber es gibt keine
Möglichkeit mehr, es abzuspielen.

#Obj
Das zu benutzende Objekt.

FileName
Die volle Pfadangabe zum IFF Sound.

Result
Dies ergibt TRUE, wenn das Sound Objekt mit dem angegebenen
IFF Sound initialisiert werden konnte, andernfalls FALSE.
```

1.13 peeksounddata

```
  SYNTAX
Result.b = PeekSoundData(#Obj.w,Position.l)

  FUNCTION
Diese Funktion liest einige Sound Daten von einem Sound Objekt
aus.

#Obj
Das zu benutzende Objekt.

Position
Gibt an, wo im Sound Objekt die Daten ausgelesen werden sollen.

Result
Die ausgelesenen Daten, sie können zwischen -128 und 127 liegen.
```

1.14 playsound

```
  SYNTAX
Result.w = PlaySound(#Obj.w,Repeat.w)

  FUNCTION
Diese Funktion spielt das angegebene Sound Objekt ab. Dabei
werden die im Objekt gespeicherten Werte für Periode,
Lautstärke und Kanäle benutzt.

#Obj
Das abzuspielende Objekt.

Repeat
Soll der Sound ein- oder mehrmals abgespielt werden, setzen Sie
diesen Parameter auf einen positiven Wert (ungleich null). Soll
der Sound endlos wiederholt werden, geben Sie einen negativen
Wert an.

Result
Diese Maske zeigt die aktuellen Kanäle an, welche für den Sound
benutzt werden. Null bedeutet, dass der Sound nicht abgespielt
wird.
```

1.15 pokesounddata

```
  SYNTAX
PokeSoundData(#Obj.w,Position.l,Data.b)

  STATEMENT
Dieser Befehl schreibt einige Daten in ein Sound Objekt.

#Obj
Das zu benutzende Objekt.
```

```
Position
Gibt an, wo im Sound Objekt die Daten geschrieben werden sollen.

Data
Die zu schreibenden Daten, sollte reichen von -128 bis 127.
```

1.16 savesound

```
  SYNTAX
Result.b = SaveSound(#Obj.w,FileName$)

  FUNCTION
Diese Funktion speichert ein Sound Objekt als einen IFF Sound
auf Disk.

#Obj
Das zu speichernde Objekt.

FileName
Die volle Pfadangabe, wo der IFF Sound gespeichert werden soll.

Result
Ergibt TRUE, wenn das Sound Objekt erfolgreich als ein IFF Sound
auf Disk gespeichert werden konnte, andernfalls FALSE.
```

1.17 setsoundchannels

```
  SYNTAX
SetSoundChannels(#Obj.w,Channels.w)

  STATEMENT
Dieser Befehl legt die Kanäle im Sound Objekt fest, welche
beim Abspielen des Sounds genutzt werden sollen.

Die Kanäle werden beim Initialisieren des Sound Objekts mit
LoadSound() oder DecodeSound() auf 15 gesetzt.

#Obj
Das Objekt, bei welchem die Kanäle festgelegt werden sollen.

Channels
Dies ist eine Maske, welche die zu benutzenden Kanäle beim
angegebenen Objekt festlegt.

1 = benutze nur Kanal 0
2 = benutze nur Kanal 1
4 = benutze nur Kanal 2
8 = benutze nur Kanal 3

Wenn zusammenaddiert:
```

```
 5 = benutze Kanäle 0 und 2
15 = benutze alle Kanäle
```

1.18 setsoundperiod

```
  SYNTAX
SetSoundPeriod(#Obj.w,Period.w)

  STATEMENT
Dieser Befehl legt die Periode im Sound Objekt fest, welche
beim Abspielen des Sounds benutzt wird.

Die Periode wird automatisch auf den richtigen Wert gesetzt
(der IFF Datei entnommen), wenn das Sound Objekt mittels
LoadSound() oder DecodeSound() initialisiert wird.

#Obj
Das Objekt, für welches die Periode definiert werden soll.

Period
Die Periode; sie sollte den endgültigen Wert darstellen,
da damit keine Berechnungen erfolgen.
```

1.19 setsoundvolume

```
  SYNTAX
SetSoundVolume(#Obj.w,Volume.w)

  STATEMENT
Dieser Befehl legt die Lautstärke (Volume) im Sound
Objekt fest, welche beim Abspielen des Sounds benutzt
wird.

Die Lautstärke wird auf 64 festgelegt, wenn das Sound Objekt
mittels LoadSound() oder DecodeSound() initialisiert
wird.

#Obj
Das Objekt, bei welchem die Lautstärke festgelegt werden soll.

Volume
Die Lautstärke, sie sollte zwischen 0 und 64 liegen.
```

1.20 soundfilter

```
  SYNTAX
SoundFilter(ON/OFF)

  STATEMENT
Dieser Befehl schaltet den Audiofilter ein oder aus.
```

```
ON/OFF
Setzen Sie diesen Parameter auf TRUE, um den Audiofilter EIN
zu schalten oder auf FALSE, um den Audiofilter AUS zu schalten.
```

1.21 stopsound

```
  SYNTAX
StopSound(#Obj.w)

  STATEMENT
Benutzen Sie diesen Befehl, um einen Sound anzuhalten.

#Obj
Das anzuhaltende Objekt.
```