

# **MacSND**

Olaf `Olsen' Barthel

<b>COLLABORATORS</b>
----------------------

	<i>TITLE :</i> MacSND		
<i>ACTION</i>	<i>NAME</i>	<i>DATE</i>	<i>SIGNATURE</i>
WRITTEN BY	Olaf `Olsen` Barthel	August 25, 2024	

<b>REVISION HISTORY</b>
-------------------------

NUMBER	DATE	DESCRIPTION	NAME

# Contents

<b>1</b>	<b>MacSND</b>	<b>1</b>
1.1	MacSND sound datatype . . . . .	1
1.2	introduction . . . . .	1
1.3	installation . . . . .	1
1.4	features . . . . .	2
1.5	bugs . . . . .	2
1.6	author . . . . .	2
1.7	history . . . . .	3

# Chapter 1

## MacSND

### 1.1 MacSND sound datatype

MacSND Sound DataType for Workbench 3.0  
Written by Olaf 'Olsen' Barthel  
Public Domain

- I. Introduction
- II. Installation
- III. Features
- IV. Bugs
- V. Author
- IV. History

### 1.2 introduction

On the Apple Macintosh computer each file stored on disk may consist of two parts; one is called the resource fork and one is called the data fork. Both parts can be independently referred to. In the data fork one will for example find the text stored in text documents while additional information such as the size and position of the window in which the text was last edited will be stored in the resource fork. On the Amiga the concept closest to resource fork would be to store the 'resource fork data' in the icon of a project file.

Sounds, such as the system beep sound, are also stored in the resource fork structure of a file. They are commonly found in "snd " type resources which include the essential data necessary to replay it using the Apple Macintosh sound hardware. This data is organized in the form of a list of opcodes which control the sound synthesizer which unlike the Amiga audio hardware can both replay sampled sound and generate synthesized sounds like an analogue Moog synthesizer.

### 1.3 installation

---

The "MacSND" datatype distribution should consist of the following files:

- macsnd.datatype
- MacSND
- MacSND.info
- MacSND.guide
- MacSND.guide.info
- Source code, in archived form

Copy the file "macsnd.datatype" into the "SYS:Classes/DataTypes" drawer. The "MacSND" and "MacSND.info" files should be placed in the "DEVS:DataTypes" drawer. In order to use the datatype you will need to reboot the machine.

## 1.4 features

The "MacSND" datatype handles both files in MacBinary format and plain resource fork files. It scans the resource fork for "snd " type resources and will load the first entry to feature sampled sound.

Only the type #1 and #2 "snd " resources are supported. Stereo or synthesized data will be rejected.

Due to the ingenious design of the resource fork format (which I consider to be pretty silly) the "MacSND" datatype may take some time to locate "snd " resources. If you ever wondered why Apple Macintosh applications spend so much time seeking back and forth on the media in order to load data: it's the resource fork format which requires it.

The "MacSND" datatype will read and process any file in resource fork format, such as applications which may include embedded "snd " resources.

## 1.5 bugs

When running under Workbench 3.0, sounds larger than approximately 102400 bytes will not play correctly. This is actually not a fault of the macsnd.datatype, but rather of the sound.datatype superclass which does no double-buffering. Application software can, however, easily compensate for this limitation.

## 1.6 author

The "MacSND" datatype was written by Olaf 'Olsen' Barthel using SAS/C 6.51. In order to recompile it you will need the Includes Release 40.15 or later. The datatype, the documentation and the accompanying source code are placed in the public domain.

---

The author can be reached at:

Olaf Barthel  
Brabeckstraße 35  
D-30559 Hannover  
Federal Republic of Germany

or via Internet:

olsen@sourcery.han.de

## 1.7 history

v1.7

Small changes to the sound normalization code.

v1.6

This release no longer requires v40, but you may still experience problems when running under v39. See the section entitled Bugs for more information. There is no longer so much stack space required to use this datatype. If necessary it will allocate new stack memory for itself and use it. Older releases could slightly distort the sound while normalizing it, this has been fixed.