# makelib

**COLLABORATORS**

| | *TITLE* :<br><br>makelib | |
|---|---|---|

| *ACTION* | *NAME* | *DATE* | *SIGNATURE* |
|---|---|---|---|
| WRITTEN BY | | August 8, 2022 | |

**REVISION HISTORY**

| NUMBER | DATE | DESCRIPTION | NAME |
|---|---|---|---|
| | | | |

# Contents

**1 makelib** **1**

1.1 makelib - Dokumentation . . . 1
1.2 makelib - Einführung . . . 1
1.3 makelib - Installation . . . 2
1.4 makelib - Konfiguration . . . 2
1.5 makelib - Argumente . . . 3
1.6 makelib - Zu beachten . . . 4
1.7 makelib - Autor & Adresse . . . 4
1.8 makelib - Urheberrecht & Distribution . . . 5
1.9 makelib - History . . . 5
1.10 makelib - Documentation . . . 5
1.11 makelib - Introduction . . . 6
1.12 makelib - Installation . . . 7
1.13 makelib - Configuration . . . 7
1.14 makelib - Arguments . . . 8
1.15 makelib - Remarks . . . 8
1.16 makelib - Author & Address . . . 9
1.17 makelib - Copyright Notice & Distribution . . . 9
1.18 makelib - History . . . 10

# Chapter 1

# makelib

## 1.1 makelib - Dokumentation

```
                makelib 1.0 (29.6.1999)

Dieses Dokument beinhaltet die folgenden Kapitel:


                 Einführung
                 Was ist es? Und wozu brauche ich es?

                 Installation

                 Argumente
                 Wie benutze ich es?

                 Konfiguration
                 Wie sage ich ihm, was es tun soll?

                 Zu beachten
                 Wieso tut es nicht?

                 Autor
                 Mit Kontaktadresse

                 Urheberrecht
                 Rechtliches & Hinweise zur Distribution

                 History
                Außerdem enthält es zusätzlich die Übersetzung obiger Texte in:

                 English
                Copyright ©1999 pinc Software. Alle Rechte vorbehalten.
```

## 1.2 makelib - Einführung

"makelib" ist ein kleines Programm, daß aus dem C-Source eines Library-
Projektes die Funktionsköpfe herausextrahiert und daraus folgende
Dateien automatisch erstellt:

- clib/mylib_protos.h
- pragmas/mylib_pragmas.h
- proto/mylib.h
- mylib_lib.fd

Mal abgesehen von dem erheblichen Wartungsaufwand, den dieses Programm
spart, sorgt es damit auch für die Vermeidung von Fehlern und ein ein-
heitliches Aussehen (mit Kommentaren) der obigen Dateien.

Die erstellten Dateien sind kompatibel mit den Original-Includes von
Commodore bzw. Amiga Inc.. D.h. einige C-Compiler könnten Probleme
mit ihnen haben, da einige SAS/C-Spezialitäten benutzt werden (diverse
#pragma-Anweisungen).

Andere Programmiersprachen können nur zum Teil eingebunden werden; die
Funktionsköpfe kann man auch in der
Konfigurationsdatei
definieren -
den Library-Funktionsvektor allerdings nicht.
Da makelib nur C-Kommentare versteht, könnte man da vielleicht noch
ein wenig tricksen.

## 1.3 makelib - Installation

Zur Installation wird das Programm einfach z.B. ins C-Verzeichnis kopiert
(sinnvollerweise ein im Pfad liegendes Verzeichnis).

Zur Konfiguration wird eine
eigene Datei
für jedes Projekt benutzt.

"makelib" setzt mindestens OS 3.0 voraus, sollte aber auch mit höheren
Versionen funktionieren.

## 1.4 makelib - Konfiguration

Die eigentliche Konfiguration von makelib wird über eine eigene Datei für
jedes Projekt geregelt. Voreingestellt bezieht sich makelib auf "makelib.with"
im aktuellen Verzeichnis.

In einer Konfigurationsdatei wird zwischen zeilenabhängigen und globalen
Kommandos unterschieden, die beide mit einem Punkt ('.') in der ersten
Spalte eingeleitet werden (z.B. ".bias 30").

Außerdem werden die Kommandos case-sensitive behandelt, sie müssen also
klein geschrieben werden.

Es gibt folgende globale Kommandos:

base Name Name der Library-Base, z.B. MyLibBase

bias Offset Dezimaler Offset zur ersten richtigen Funktion.
Voreingestellt ist 30.

func Prototype C-Prototype einer Funktion, die z.B. nur in Assembler
vorliegt.

libvectors Symbol Legt das C-Symbol fest, das für den Library-
Funktionsvektor benutzt wird.
Voreingestellt ist "LibVectors".

table Symbol Synonym für libvectors.

decimals Legt fest, daß die Funktionsoffsets als Dezimal-
zahlen interpretiert werden (sonst Hexadezimal).

protoheader Ein "Prolog" für die mylib_protos.h-Datei, in der
bspw. benötigte C-Header eingebunden werden.
Die folgenden Zeilen bis zum nächsten Kommando werden
diesem Kommando zugedacht.

Bei den zeilenabhängigen Kommandos steht nach dem Punkt erstmal eine
Hexadezimalzahl (Dezimalzahl mit decimals), die angibt, um welchen
Offset es geht. Diese Angaben sind mit den (positiven) LVOs bzw. den Zahlen
in der #pragma-Anweisung in den pragmas-Dateien identisch.
Eine solche Zeile sieht z.B. so aus: ".36 private mylibprivate1".

private Symbol Definiert eine private Funktion, das angegebene
Symbol wird in der fd-Datei benutzt.

tagcall Symbol Fügt zur eigentlichen Funktion ein #pragma tagcall
mit dem angegebenen Symbol ein.

rem Fügt den Text in den folgenden Zeilen als Kommentar
in die jeweiligen Dateien ein.

Im beigefügten Beispiel können die meisten dieser Kommandos in der Praxis
begutachtet werden.
Falls die Kommentare in den Dateien mal nicht so gut aussehen, es gibt
Editoren (wie z.B. den GoldED von Dietmar Eilert), die die Leerzeichen
am Ende einer Zeile automatisch entfernen.

## 1.5 makelib - Argumente

Die Argumente werden bei makelib benutzt, um Datei-spezifische ↩
Dinge
festzulegen. Für inhaltliche Dinge ist die
Konfigurationsdatei

```
                zu benutzen.

W=WITH/K    Legt die zu benutzende Konfigurationsdatei fest. Standardmäßig
            wird "makelib.with" im aktuellen Verzeichnis benutzt.

FILE/A/M    Die nach Funktionsköpfen, Prototypen und Library-Einsprungtabellen
            zu durchsuchenden Dateien (Muster erlaubt).

ALL/S       auch Unterverzeichnisse werden durchsucht.

TO/K        gibt das Basisverzeichnis für die zu erstellenden Dateien an.
            Z.B. "include:". Funktioniert nur, wenn NAME angegeben ist.

NAME/K      legt den Basis-Namen der zu erstellenden Dateien fest, z.B.
            "mylib". Sorgt grundsätzlich dafür, daß die Dateien in "clib/",
            "pragmas/" und "proto/" angelegt werden.

FD/K        legt den Namen der .fd-Datei an. Z.B. "fd:mylib_lib.fd". Die
            Datei wird nur erstellt, wenn diese Option gesetzt ist.

V=VERBOSE/S Gibt ein paar zusätzliche Informationen aus.

Q=QUIET/S   Gibt keinen Versionstext aus (Fehler und Warnungen werden nicht
            unterdrückt.

PROTOS/S    Legt fest, daß nicht nur Funktionsköpfe, sondern auch Prototypes
            gescannt werden sollen.


Ein Beispielaufruf könnte so aussehen:

makelib gtdrag_includes.h gtdrag_lib.c to=include name=gtdrag fd=gtdrag_lib.fd  ↩
   protos
```

## 1.6 makelib - Zu beachten

```
Funktionsköpfe müssen in einer Zeile stehen und die Argumente müssen
  auch bei Prototypes benannt sein, damit makelib damit arbeiten kann
  (andernfalls wird gewarnt).
  Bislang werden die C-Schlüsselwörter "static" und "const" nicht erkannt,
  das kann sich aber bei diesbezüglichen Problemen durchaus noch ändern.

Library-Vektoren (Arrays) werden erkannt, wenn das angegebene Symbol
  gefolgt von "[]", einer beliebigen Anzahl an White-Spaces und einem "="
  in einer Zeile steht (im äußersten Block).
  Die Elemente im folgenden Block werden als Definition dieses Library-Vektors
  benutzt.
  Im beigefügten Beispiel ist eine gültige Defition enthalten.
```

## 1.7 makelib - Autor & Adresse

Die Adresse für Bug-Reports, Ideen und andere Dinge:

Per Post:
  pinc Software / Axel Dörfler
  Im Wiesengrund 3
  49205 Hasbergen

eMail-Adresse:
  axeld@bigfoot.de (Axel Dörfler)
  (adoerfle@rz.uni-osnabrueck.de)

WWW:
  www-lehre.informatik.uni-osnabrueck.de/~adoerfle/
  www.pinc-software.de/

Postkarten und andere Vergütungen sind natürlich erwünscht.

## 1.8 makelib - Urheberrecht & Distribution



## 1.9 makelib - History

Version 1.0 (29.6.1999)
 · erste Version

## 1.10 makelib - Documentation

makelib 1.3 (29.6.1999)

Table of Contents:

Introduction
What's all about?

Installation

Arguments
how to use it

Configuration
dto. :-)

Remarks
why it doesn't work

Author
with address

Copyright Notice

History
This document contains furthermore the following translations of the above:

Deutsch


## 1.11 makelib - Introduction

"makelib" is a small tool which extracts function headers and prototypes
out of standard ANSI-C-Source and automatically creates the following
files (replace mylib with your one of your project names):

- clib/mylib_protos.h
- pragmas/mylib_pragmas.h
- proto/mylib.h
- mylib_lib.fd

This reduce maintenance of these files and dependent errors to a
minimum. Furthermore, it provides a uniform look with the same
comments for all files.

The created files are compatible to the original includes made by
Commodore/Amiga Inc.. This includes SAS/C specific behaviour (several
#pragma commands).

Other programming languages may be used only in parts: the function
headers may be put into the
configuration file
, the library vector
must not.
Since makelib only supports C-comments, you may find a way to work
around this.

## 1.12 makelib - Installation

To install the programme you simply copy it in the drawer you want it
to have. It's quite useful to choose one which is in the system path
like C:.

To use this programme within one of your projects, you also need to
make a configuration file for it.

The makelib requires OS 3.0 or higher. I test and developed it under
OS 3.0 where it runs perfectly (at least on my system :-).

## 1.13 makelib - Configuration

makelib will be configured with a special file for every project you use
it for. This file defaults to be "makelib.with" in the working directory.

They are two different kinds of commands, the global ones and the line-
dependent ones. Both of them are introduced with a point ('.') in the
first row of a line (e.g. ".bias 30").
Furthermore, the commands are scanned case-sensitive; they must be written
in lower case.

There are the following global commands:

| | |
|---|---|
| base name | Name of the Library-Base, e.g. MyLibBase |
| bias offset | Decimal offset to the first function. Defaults to 30. |
| func prototype | C-Prototype of a function which is implemented in e.g. assembler only. |
| libvectors symbol | Specifies the C-symbol which is used for the library function array. Defaults to "LibVectors". |
| table symbol | Synonym for libvectors. |
| decimals | If set, the function offset (in this file) are interpreted as decimals rather than hexadecimals. |
| protoheader | An "Introduction" for the mylib_protos.h-file in which you may include C-Headers or the like. The following lines to the next command will be used for this. |

The line-dependent commands are preceded by a hexadecimal (or a decimal, if
the decimals command is set) number representing the function offset.
This number is identical to the LVO or the number in a #pragma-command.
Such a command may look like this: ".36 private mylibprivate1".

| | |
|---|---|
| private symbol | Defines a private function; the symbol will be used in the fd-file. |
| tagcall symbol | Adds a "#pragma tagcall" with the symbol to the normal |

```
                  function.

rem               Inserts the following lines as comments in every file.


You may have a look at the includes example which makes use of most of the
the existing commands.
If your comments doesn't look nice in the created files, it may your
editor which cuts the succeding blanks of a line (e.g. Dietmar Eilert's
GoldED).
```

## 1.14 makelib - Arguments

```
                "makelib" use the command line arguments to specify file-dependent ↩
                    things.
You use the
                configuration-file
                 for content-dependent things.

W=WITH/K    Specifies the configuration file. Defaults to "makelib.with"
            in the current directory.

FILE/A/M    The files being searched for function-headers, prototypes and
            library-vectors (patterns allowed).

ALL/S       scans through sub-directories.

TO/K        specifies the base directory for the files being created, e.g.
            "include:". Works only if NAME is specified.

NAME/K      specifies the base name for the files being created, e.g.
            "mylib". Enables the creation of files in "clib/", "pragmas/",
            and "proto/".

FD/K        specifies the name of the .fd-file, e.g. "fd:mylib_lib.fd". This
            file will be created only, if this option is set.

V=VERBOSE/S prints out additional informations.

Q=QUIET/S   Suppresses the version banner. (Errors and warnings are printed
            further on.)

PROTOS/S    Set this flag if you want makelib to scan prototypes, too.


An example call may look like this:

makelib gtdrag_includes.h gtdrag_lib.c to=include name=gtdrag fd=gtdrag_lib.fd  ↩
   protos
```

## 1.15 makelib - Remarks

```
Function-headers must fit in one line. The arguments must be
  named even at prototypes to let makelib work with them (otherwise
  you will be warned).
  The C-keywords "static" and "const" aren't yet supported; this may
  change if you have problems with it.

Library-vectors (arrays) are recognised if the specified symbol
  followed by "[]", a white-space and a "=" are in one row (in the outer
  block).
  The elements in the following block are presumed to be the library symbols
  used by this vector.
  You may have a look at this example where you can find a valid definition.
```

## 1.16 makelib - Author & Address

```
The Address for bug-reports, ideas and other hopefully useful things:

Via snail-mail:
  pinc Software / Axel Dörfler
  Im Wiesengrund 3
  49205 Hasbergen

eMail-Address:
  axeld@bigfoot.de (Axel Dörfler)
  (adoerfle@uni-osnabrueck.de)

WWW:
  www-lehre.informatik.uni-osnabrueck.de/~adoerfle/
  www.pinc-software.de/

If you want to send small donations or cards - be welcome :)
```

## 1.17 makelib - Copyright Notice & Distribution

## 1.18 makelib - History

```
Version 1.0 (29.6.1999)
 · initial version
```