DependantScan

Ray Darrah III

COLLABORATORS			
	TITLE : DependantScan		
ACTION	*NAME*	*DATE*	*SIGNATURE*
WRITTEN BY	Ray Darrah III	October 9, 2022	

REVISION HISTORY			
NUMBER	DATE	DESCRIPTION	NAME

Contents

Chapter 1

DependantScan

1.1 DependantScan.guide

DependantScan V1.1 - a MakeFile maintenance utility for Amiga computers

1.2 dependantscan introduction

DependantScan is a program for use by Amiga C programmers. If you do not program in C, DependantScan will be of little use to you.

The DependantScan utility creates MakeFiles . DependantScan examines the C source code files in a project to determine which header files the source code files are dependant on, and generates a MakeFile with the determined relationships.

With DependantScan, a C developer no longer needs to edit their MakeFiles after a changing the source code to include another header file. Instead, the C developer merely runs DependantScan and the MakeFile containing the new dependency information is automatically generated.

This program was written as a side project when I became tired of manually maintaining my own MakeFiles. I tried several utilities from the AmiNet and the mkmk utility from SAS/C, but none of them worked as I wanted. And thus, DependantScan was born.

1.3 aminet

The AmiNet is a portion of the Internet which contains gigabytes of free Amiga software for the taking. If you don't have access to the Internet, the software can also be purchased on CD-ROMs from various Amiga software vendors.

1.4 makefile

If you write in C and do not use MakeFiles, use them.

With regard to C programming, the most common use of MakeFiles (or in SAS/C's case SmakeFiles) are to only compile those source code modules that need to be compiled. For example, if there are 10 source code files in a project and you edit one of them, then just that one file should be re-compiled thus saving the compile time of the other nine source code modules.

In a similar manner, if you make a change to one of the header files #included by a source code module, the source code module should also be re-compiled.

DependantScan automates the process of creating and maintaining MakeFiles. Using MakeFiles generated by DependantScan can be easier than not using MakeFiles at all.

It can be dangerous not to use MakeFiles! Consider the case where a programmer edits a header file but does not re-compile all of the source code modules that use the header file. If one of the non-compiled source code modules uses some of the edited data, the result is a program that in most cases will behave strangely. This can easily happen when using the MAKE switch of SAS/C or using SAS/C's Build icon with no MakeFile.

The name of the MakeFile created by DependantScan is controlled with the Makefile/S and MAKEFILE= options.

The MakeFiles created by DependantScan use relative file paths and are designed to be processed with the projects path as the current directory.

1.5 features & limitations

* DependantScan is Executable from the WorkBench or the CLI. All command-line options are available as tool types in an icon.

* Although written for use with SAS/C, the MakeFiles created by DependantScan may (but I doubt it) be compatible with DICE, UNIX make, and/or Manx C.

* Only a single project and set of compile options are allowed per MakeFile.

* DependantScan does not scan assembly files. But, assembly object files can be included in a project with the use of the FROM keyword/tool type.

* #include files (header files) that are included by other header files are not supported.

* Projects with source code in multiple directories must conform to the proper development model .

* DependantScan creates the MakeFile with the assumption that the linker supports the FROM, WITH, TO and LIBRARY keywords in the same manner as SAS/C's development system. If your C development system doesn't support these keywords, you will have to either edit the source code or contact the author concerning how your linker works.

Other limitations can be found in the Development Model . Please contact the author with your favorite(?) limitation and he will consider removing it.

1.6 dependantscan installation

@ keywords install installation

Installation of DependantScan can be accomplished with the following steps:

* Copy the DependantScan program to some place in your path .

* If you prefer to use the icon-based compilation system devised by SAS/C, I suggest that you move the Scanner.info example project to your sc:starter_project directory. That way, the scsetup program will automatically put the icon in your new project directories.

* The archive is an image of this branch of my development tree and therefore can be copied as-is to your development tree if you wish to preserve/use the source code .

1.7 development model

In order for DependantScan to work properly, certain restrictions must be placed upon your C development system and the way you code. I hope that most people will not find these restrictions too limiting.

* All the source code files necessary to build a project must appear to be in the same directory - or in sub-directories off of a directory. The AmigaDOS MakeLink command may be used to have it appear as if source code files from other directories actually appear in the same directory.

* All #include statements in source code files that reference standard Amiga header files or compiler header files should be in angle brackets. Example: #include <stdio.h>.

* All other #include statements must use quotation marks. Example #include "DependantScan.h".

* DependantScan will search sub-directories off of the root path . So, any source code modules found in these sub-directories will also be included in the MakeFile .

See also Features & Limitations .

1.8 options

All of DependantScan options are available either command-line parameters or Tool Types.

FilesOnLine/N/K FILESONLINE= - number of files per MakeFile line

From/K FROM= - linker object files

Library/K LIBRARY= - linker libraries

MakeFile/K MAKEFILE= - name of the MakeFile

Match/K MATCH= - files to be scanned

Object_Dir/K OBJECT_DIR= - directory for the object files

Path/K PATH= - project directory

Project/K PROJECT= - executable name

Rules/K RULES= - file to be put into the MakeFile

Verbose/S VERBOSE - progress display switch

1.9 filesonline option

N=FilesOnLine/N/K - as a CLI argument

FILESONLINE= - as a Tool Type (from WorkBench)

Default Examples:

N 5 - from the CLI

filesonline 5 - from the CLI

FILESONLINE=5 - from the WorkBench

This optional parameter is used primarily to change the human readability of the MakeFile . It sets the maximum number of dependant files that appear on any one line of the MakeFile.

1.10 from option

From/K - as a CLI argument

FROM= - as a Tool Type (from WorkBench)

Default Examples:

from "LIB:c.o" - from the CLI

FROM=LIB:c.o - from the WorkBench

This optional parameter changes what is put after the linker's FROM keyword in the link section of the MakeFile . If a project has non-standard startup code or no startup code at all, this parameter will have to be specified. Also, this parameter may be used to include object files that are not compiled with the output MakeFile.

See also: Features & Limitations

1.11 library option

LIB=Library/K - as a CLI argument

LIBRARY= - as a Tool Type (from WorkBench)

Default Examples:

lib "lib:sc.lib + lib:amiga.lib" - from the CLI

library "lib:sc.lib + lib:amiga.lib" - from the CLI

LIBRARY=lib:sc.lib + lib:amiga.lib - from the WorkBench

This optional parameter specifies the first text that follows the linker's LIBRARY keyword. It is typically used to change the libraries that a project is linked with.

See also: Features & Limitations

1.12 makefile option

MK=MakeFile/K - as a CLI argument

MAKEFILE= - as a Tool Type (from WorkBench)

Default Examples:

mk "SMakeFile" - from the CLI

makefile SMakeFile - from the CLI

MAKEFILE=SmakeFile - from the WorkBench

This optional parameter specifies the name of the MakeFile that DependantScan will create.

1.13 match option

Match/K - as a CLI argument

MATCH= - as a Tool Type (from WorkBench)

Default Examples:

match "#?.c" - from the CLI

MATCH=#?.c - from the WorkBench

This optional parameter specifies the files to be scanned by {u}DependantScan. The program does not need to scan the header files in your development model .

1.14 object_dir option

Obj=Object_Dir/K - as a CLI argument

OBJECT_DIR= - as a Tool Type (from WorkBench)

Default Examples:

obj "Object/" - from the CLI

object_dir Object/ - from the CLI

OBJECT_DIR=Object/ - from the WorkBench

This optional parameter specifies the directory where the object files will be located for the compile. A trailing slash (/) or colon (:) is required. The default listed above is slightly misleading because if this option is not specified, DependantScan will attempt to retrieve the value from the SCOPTIONS file in the project's path . The SCOPTIONS file is usually edited with the SCOpts program (the scoptions icon) in the SAS/C development model .

It is recommended that you put object files compiled with different options in different directories. The SCOptions.Normal and SCOptions.Debug files in this archive demonstrate how you can use the IconJ program to easily switch between different compile options.

1.15 path option

Path/K - as a CLI argument

PATH= - as a Tool Type (from WorkBench)

Examples:

path Work:C - from the CLI

PATH=Work:C - from the WorkBench

This optional parameter specifies the root directory of the development model i.e. where to look to find source code files. The default for this option is the current directory (from the CLI) or the directory belonging to the icon (when ran from the WorkBench). DependantScan will process all files that match the Match/K MATCH= file specification in this directory and it's sub-directories.

1.16 project option

Project/K - as a CLI argument

PROJECT= - as a Tool Type (from WorkBench)

Default Examples:

project "Executable" - from the CLI

PROJECT=Executable - from the WorkBench

This optional parameter specifies the primary object that the MakeFile will make. If this option is not specified, DependantScan will attempt to retrieve the value from the SCOPTIONS file in the project's path . If the parameter cannot be retrieved from SCOPTIONS file, the default will be "Executable", as illustrated above. Typically, this is the name of the program that you are working on.

1.17 rules option

Rules/K - as a CLI argument

RULES= - as a Tool Type (from WorkBench)

Examples:

rules "Compile.rules" - from the CLI

RULES=Compile.rules - from the WorkBench

This optional parameter specifies the name of a file to be inserted at the top of the MakeFile . By default, no file will be inserted. This may be useful if you are using a different compiler and/or make program.

1.18 verbose switch

Verbose/S - as a CLI argument

VERBOSE - as a Tool Type

Examples:

verbose - from the CLI

VERBOSE - from the WorkBench

This optional switch is supplied when you want DependantScan to display additional information about the process and to present a requester at the end of the operation.

1.19 source code

The DependantScan source code supplied in this archive is only availble from the CLI. It can be found in the Source directory.

As supplied, you should be able to re-compile DependantScan by clicking on the SCOptions.Normal or SCOptions.Debug icons to select the compile mode and then by typing Smakeg from the CLI while in the source directory.

Information about how to make an application's source code compatible with DependantScan can be found in the Development Model section.

If you have further difficulties with the program, feel free to contact the author .

1.20 iconj program

The IconJ program is a public domain utility that acts very much like the IconX program that comes with AmigaDOS. However, the IconJ program can read the script lines to execute from the tool types of an icon. The SCOptions.Normal and SCOptions.Debug icons provided demonstrate how one can use this utility to quickly switch between compile options.

1.21 legal stuff

Verbatim copies of this program and documentation may be distributed freely. All changes to this program and or documentation must be approved by the author prior to distribution.

No guarantee of any kind is given that the program described in this document is error-free. The user assumes all risk for use or non-use of this program.

See also: Standard Disclaimer .

1.22 registration

If you find this program and/or source code helpful - and use it on a regular basis - you are encouraged, but not required, to send the author some currency or a gift.

If you should decide to send the author something, use your ability to pay, your conscience and the program's usefulness as your guide as to the amount.

See also: Standard Disclaimer .

1.23 ray darrah iii

Ray Darrah is a software engineer who unfortunately professionally programs the worlds most disgusting computer system: the IBM-PC/AT running MS-DOS/Windoze. Ever since his purchase of one of the original A1000's in 1986, Ray's one computing love has always been the true 32-bit real multitasking system known as AmigaOS.

The author can be reached via snail mail at:

Ray Darrah III

P.O. Box 2618

Redmond WA, 98073

U.S.A.

OR on the Internet at:

raydar@netcom.com

1.24 standard disclaimer

*** STANDARD DISCLAIMER ***

This product is meant for educational purposes only. Any resemblance to real persons, living or dead is purely coincidental. Void where prohibited. Some assembly required. List each check separately by bank number. Batteries not included. Contents may settle during shipment. Use only as directed. No other warranty expressed or implied. Do not use while operating a motor vehicle or heavy equipment. Postage will be paid by addressee. Subject to CAB approval. This is not an offer to sell securities. Apply only to affected area. May be too intense for some viewers. Do not stamp. Use other side for additional listings. For recreational use only. Do not disturb. All models over 18 years of age. Proof of age on file. If condition persists, consult your physician. No user-serviceable parts inside. Lather, rinse, repeat. Freshest if eaten before date on carton. Subject to change without notice. Times approximate. Simulated picture. No postage necessary if mailed in the United States. Breaking seal constitutes acceptance of agreement. For off-road use only. As seen on TV. One size fits all. Many suitcases look alike. For external use only. Contains a substantial amount of non-tobacco ingredients. Colors may, in time, fade. We have sent the forms which seem to be right for you. Slippery when wet. For office use only. Not affiliated with the American Red Cross. Drop in any mailbox. Edited for television. Keep cool; process promptly. Post office will not deliver without postage. List was current at time of printing. It has been modified to fit your screen. Return to sender, no forwarding order on file, unable to forward. Not responsible for direct, indirect, incidental or consequential damages resulting from any defect, error or failure to perform. At participating locations. Not the Beatles. Penalty for private use. See label for sequence. Substantial penalty for early withdrawal. Do not write below this line. Falling rock. Lost ticket pays maximum rate. Your cancelled check is your receipt. Add toner. Place stamp here. Avoid contact with skin. Sanitized for your protection. Be sure each item is properly endorsed. Sign here without admitting guilt. Slightly higher west of the Mississippi. Employees and their families are not eligible. Beware of dog. Contestants have been briefed on some questions before the show. Limited time offer, call now to insure prompt delivery. You must be present to win. It is a violation of Federal law to use this product in a manner inconsistent with its labeling. No passes accepted for this engagement. No purchase necessary. Processed at location stamped in code at top of carton. Shading within a garment may occur. Use only in well-ventilated area. Keep away from fire or flame. Replace with same type. Approved for veterans. Booths for one person only. Check here if tax deductible. Some equipment shown is optional. Price does not include taxes. No Canadian coins. Not recommended for children. Prerecorded for this time zone. Reproduction strictly prohibited.

No solicitors. No alcohol, dogs, or horses. No anchovies unless otherwise specified. Restaurant package, not for resale. List at least two alternate dates. First pull up, then pull down. Call toll free before digging. Driver does not carry cash. Some of the trademarks mentioned in this product appear for identification purposes only. Record additional transactions on back of previous stub. Actual mileage will vary. Decision of judges is final.

This supersedes all previous notices.