



[\[Home\]](#) [\[Search\]](#) [\[D\]](#)

Last update Feb 5, 2004

Compiler for D Programming Language

- D for [Win32](#)
- D for [x86 Linux](#)
- [general](#)

Files Common to Win32 and Linux

\dmd\src\phobos\
 D runtime library source

\dmd\src\dmd\
 D compiler front end source under dual (GPL and Artistic) license

\dmd\html\d\
 Documentation

\dmd\samples\d\
 Sample D programs

Win32 D Compiler

Files

\dmd\bin\dmd.exe
 D compiler executable

\dmd\bin\shell.exe
 Simple command line shell

\dmd\bin\sc.ini
 Global compiler settings

\dmd\lib\phobos.lib
 D runtime library

Requirements

- 32 bit Windows operating system
- [D compiler](#) for Win32
- [linker and utilities](#) for Win32

Installation

Unzip the files in the root directory. It will create a \dmd directory with all the files in it. All the tools are command line tools, which means they are run from a console window. Create a console window in Windows XP by clicking on [Start][Command Prompt].

Example

Run:

```
\dmd\bin\shell all.sh
```

in the \dmd\samples\d directory for several small examples.

Compiler Arguments and Switches

dmd *files...* *-switch...*

files...

Extension	File Type
<i>none</i>	D source files
.d	D source files
.obj	Object files to link in
.exe	Name output executable file
.def	module definition file
.res	resource file

-c

compile only, do not link

-d

allow deprecated features

-debug

compile in debug code

-debug=*level*

compile in debug code \leq *level*

-debug=*ident*

compile in debug code identified by *ident*

-g

add symbolic debug info

-gt

add trace profiling hooks

-inline

inline expand functions

-I*path*

where to look for imports. *path* is a ; separated list of paths. Multiple **-I**'s can be used, and the paths are searched in the same order.

-L*linkerflag*

pass *linkerflag* to the linker, for example, `/ma/li`

-O

optimize

-odobjdir

write object files relative to directory *objdir* instead of to the current directory

-of*filename*

set output file name to *filename* in the output directory

-op

normally the path for **.d** source files is stripped off when generating an object file name. **-op** will leave it on.

-release

compile release version

-unittest

compile in unittest code

-v

verbose

-version=*level*

compile in version code \geq *level*

-version=*ident*

compile in version code identified by *ident*

Linking

Linking is done directly by the **dmd** compiler after a successful compile. To prevent **dmd** from running

the linker, use the **-c** switch.

The programs must be linked with the D runtime library **phobos.lib**, followed by the C runtime library **snn.lib**. This is done automatically as long as the directories for the libraries are on the LIB environment variable path. A typical way to set LIB would be:

```
set LIB=\dmd\lib;\dm\lib
```

Environment Variables

The D compiler dmd uses the following environment variables:

DFLAGS

The value of **DFLAGS** is treated as if it were appended to the command line to **dmd.exe**.

LIB

The linker uses LIB to search for library files. For D, it will normally be set to:

```
set LIB=\dmd\lib;\dm\lib
```

LINKCMD

dmd normally runs the linker by looking for **link.exe** along the **PATH**. To use a specific linker instead, set the **LINKCMD** environment variable to it. For example:

```
set LINKCMD=\dm\bin\link
```

PATH

If the linker is not found in the same directory as **dmd.exe** is in, the **PATH** is searched for it.

Note: other linkers named **link.exe** will likely not work. Make sure the Digital Mars **link.exe** is found first in the **PATH** before other **link.exe**'s, or use **LINKCMD** to specifically identify which linker to use.

SC.INI Initialization File

dmd will look for the initialization file **sc.ini** in the same directory **dmd.exe** resides in. If found, environment variable settings in the file will override any existing settings. This is handy to make **dmd** independent of programs with conflicting use of environment variables.

Environment variables follow the [Environment] section heading, in name=value pairs. Comments are lines that start with ;. For example:

```

; sc.ini file for dmd
; Names enclosed by %% are searched for in the existing
environment
; and inserted. The special name %@P% is replaced with the
path
; to this file.
[Environment]
LIB="%@P%\..\lib";\dm\lib
DFLAGS="-I%@P%\..\src\phobos"
LINKCMD=" %@P%\..\..\dm\bin"

```

Linux D Compiler

Files

```

/dmd/bin/dmd
    D compiler executable
/dmd/bin/dumpobj
    Elf file dumper
/dmd/bin/obj2asm
    Elf file disassembler
/dmd/bin/dmd.conf
    Global compiler settings (copy to /etc/dmd.conf)
/dmd/lib/libphobos.a
    D runtime library (copy to /usr/lib/libphobos.a)

```

Requirements

- 32 bit x86 Linux operating system
- [D compiler](#) for Linux
- Gnu C compiler (gcc)

Installation

1. Unzip the archive into your home directory. It will create a ~/dmd directory with all the files in it. All the tools are command line tools, which means they are run from a console window.
2. Edit the file ~/dmd/bin/dmd.conf to put the path in to where the phobos source files are.

3. Copy `dmd.conf` to `/etc`:

```
cp dmd/bin/dmd.conf /etc
```

4. Give execute permission to the following files:

```
chmod u+x dmd/bin/dmd dmd/bin/obj2asm dmd/bin/dumpobj
```

5. Put `dmd/bin` on your **PATH**, or copy the linux executables to `/usr/local/bin`

6. Copy the library to `/usr/lib`:

```
cp dmd/lib/libphobos.a /usr/lib
```

Compiler Arguments and Switches

dmd *files... -switch...*

files...

Extension	File Type
<i>none</i>	D source files
.d	D source files
.o	Object files to link in
.a	Library files to link in

-c
compile only, do not link

-d
allow deprecated features

-debug
compile in debug code

-debug=*level*
compile in debug code \leq *level*

-debug=*ident*
compile in debug code identified by *ident*

-g
add symbolic debug info

-gt

add trace profiling hooks (not supported under linux)

-inline

inline expand functions

-I*path*

where to look for imports. *path* is a ; separated list of paths. Multiple **-I**'s can be used, and the paths are searched in the same order.

-L*linkerflag*

pass *linkerflag* to the linker, for example, **-M**

-O

optimize

-odobjdir

write object files relative to directory *objdir* instead of to the current directory

-of*filename*

set output file name to *filename* in the output directory

-op

normally the path for **.d** source files is stripped off when generating an object file name. **-op** will leave it on.

-release

compile release version

-unittest

compile in unittest code

-v

verbose

-version=*level*

compile in version code \geq *level*

-version=*ident*

compile in version code identified by *ident*

Linking

Linking is done directly by the **dmd** compiler after a successful compile. To prevent **dmd** from running the linker, use the **-c** switch.

The actual linking is done by running **gcc**. This ensures compatibility with modules compiled with **gcc**.

Environment Variables

The D compiler **dmd** uses the following environment variables:

DFLAGS

The value of **DFLAGS** is treated as if it were appended to the command line to **dmd**.

dmd.conf Initialization File

dmd will look for the initialization file **dmd.conf** in the directory `/etc`. If found, environment variable settings in the file will override any existing settings. This is handy to make **dmd** independent of programs with conflicting use of environment variables.

Environment variables follow the `[Environment]` section heading, in `name=value` pairs. Comments are lines that start with `;`. For example:

```

; dmd.conf file for dmd
; Names enclosed by %% are searched for in the existing
environment
; and inserted. The special name %@P% is replaced with the
path
; to this file.
[Environment]
DFLAGS="-I%@P%\..\src\phobos"
```

Differences from Win32 version

- String literals are read-only. Attempting to write to them will cause a segment violation.
- The configuration file is `/etc/dmd.conf`

Linux Bugs

- `-g` is not implemented, because I haven't figured out how to do it yet. **gdb** still works, though, at the global symbol level.
- The code generator output has not been tuned yet, so it can be bloated.
- Shared libraries cannot be generated.
- The exception handling is not compatible with the way `g++` does it. I don't know if this is an issue or not.

General

Bugs

These are some of the major bugs:

- The compiler sometimes gets the line number wrong on an error.
- The phobos D runtime library is inadequate.
- Need to write a tool to convert C .h files into D imports.
- Array op= operations are not implemented.
- In preconditions and out postconditions for member functions are not inherited.
- It cannot be run from the IDDE.

Feedback

We welcome all feedback - kudos, flames, bugs, suggestions, hints, and most especially donated code!
Join the fray in the [D forum](#).

Copyright (c) 1999-2004 by Digital Mars, All Rights Reserved