Windows build prerequisites using cygwin

- Edit
- Watch

Table of contents

- Tags
- Files

Page Notifications Off

Table of contents

1. This document is for historical use only
2. Hardware Requirements
3. Software Requirements
   3.1. GNU Tools for Microsoft Windows (Cygwin)
   3.2. make
   3.3. moztools
   3.4. cvs
   3.5. Compiler & Linker
   3.6. Platform SDK
   3.7. NSIS
   3.8. 7-Zip and UPX

4. Configure the Environment
   4.1. Fix cygwin path

5. Common Problems, Hints and Restrictions

Build Instructions:

- Requirements
- Source
- Configuration
- Build
- Frequently Asked Questions
- More...

This document is for historical use only

{{template.Note("This document is a guide to the build prerequisites for building the Mozilla 1.9 codebase before the introduction of the MozillaBuild system in March 2007. If you are building Firefox 3 or greater, please see the standard Windows Build Prerequisites. **For building Firefox 2, Firefox 1.5, and other Mozilla products based on pre-1.9 branches, see the Windows Build Prerequisites on the 1.7 and 1.8 Branches.**

If you are building very old versions of the Mozilla source code, from the 1.0 branch and earlier, follow the nmake build instructions.

Hardware Requirements

Mozilla takes time to build. You need a development-class system:

- Pentium III or equivalent processor, 1GHz or better, 1.5+GHz recommended

- 512 MB RAM, 1 GB recommended
- 1.5 GB NTFS or 3 GB FAT disk space, or more
- Windows NT-class operating system (NT4.0, W2K (SP4?), Windows XP and Windows Vista)

Software Requirements

The following software needs to be installed for a standard Windows build.

Mozilla may not build if some tools are installed at a path that contains spaces or other breaking characters such as pluses, quotation marks, or meta characters. The Visual C++ tools may be installed in a directory which contains spaces (the default install location is preferred).

GNU Tools for Microsoft Windows (Cygwin)

Cygwin is a Linux-like environment with GNU tools for Windows. Mozilla uses a developer set of GNU packages, which must be installed. These include gawk, make and zip utilities. GCC is not used and does not need to be installed. You must have Cygwin 1.1.6 or higher. You can have only one cygwin version installed on your operating system; concurrent installations may interfere with each other and cause errors.

Click here to run setup.exe and follow the installation instructions. The Cygwin installer wizard runs as a Windows program. One page of the wizard presents an outline view of categories and packages. You must add packages beyond the default set.

If you'll be using Cygwin only to build Mozilla, then install only the default packages and the packages needed by Mozilla. In the version of the installer current at this writing (version 2.510.2.2), the **non-default** packages are:

- zip, unzip (under Archive)
- cvs (under Devel, only if you check out from CVS, not using tarballs)
- libiconv (under Devel, only if you build the installer)
- patchutils (under Devel)
- perl (under Interpreters)
- python (under Interpreters)
- make 3.80 is no longer available, see below

If you may be using Cygwin's GNU tools for additional development, then install the editors, developer tools and interpreters, as well as anything else that looks interesting. Make sure you get all the packages Mozilla needs too, such as unzip and zip!

Here is the **complete** cygwin package list for building Mozilla. You can use it to double-check your packages, or to understand and customize the build process:

- ash -- UNIX-like command line interpreter shell (Base category)
- coreutils -- GNU core utilities (includes fileutils, install, sh-utils, and textutils) (Base category)
- cvs -- concurrent versions system (Devel category)
- diffutils -- file comparison utility (Utils category)
- findutils (Base category)
- gawk -- pattern matching language (Base and Interpretors categories)
- grep -- text search tool (Base category)
- libiconv -- character set conversion (Devel category) - libiconv2 does not contain the iconv program needed for the installer
- make 3.80 (not 3.81, see below) -- dependency analyzer for software builds (Devel category)
- patchutils -- a small collection of programs that operate on patch files (Devel category)
- perl -- a scripting language used to control parts of the build (Interpreters category)
- python -- a scripting language used to control parts of the build (Interpreters category)
- sed -- a search and replace language (Base category)
- unzip -- zip file extraction (Archive category)
- zip -- zip file creation (Archive category)

If you plan to modify the build system's configure.in files, you will need to install an editor (such as xemacs or joe) and the autoconf-2.13 (autoconf-2.5x and 2.60 doesn't work) packages as well.

For non-debug builds, it's recommended that you install the binutils package to make strip.exe available.

You may use ActivePerl from ActiveState instead of the perl package in Cygwin if you prefer.

make

make 3.80 is no longer available in the cygwin installer (version 3.81 will **not** work), so get it from paracoda or go-Mono and place the included make.exe in your %cygwin%\bin\ directory, e.g. by starting a cygwin bash shell and executing

```
cd /
tar xjf /cygdrive/c/where-your-download-is/make-3.80-1.tar.bz2
bin/make.exe --version
```

moztools

The moztools package contains binaries and libraries necessary to build Mozilla. A single zip file containing all moztools which works with all versions of MSVC is available at moztools-static.zip.

The zip file contains a single folder named *moztools* that you should preserve. Unpack the zip file wherever you want the tools to be installed; however, Mozilla may not build if you install the moztools directory inside the cygwin directory. The MOZ_TOOLS environment variable should be set to point to this moztools folder. (You may rename the folder, as long as you update $MOZ_TOOLS accordingly.)

Note: The 1.8.1 branch (Firefox 2) is compatible with the moztools-static package. Older branches (including the 1.8.0 branch) are not; see the historical build prerequisites for details.

The moztools zip file does not contain the source for the components; if you are interested in building them from scratch, you can obtain the source here. The ooo-build system for building OpenOffice.org builds the tools from scratch using this code and contains some relevant patches for anyone who doesn't want to use the binaries provided.

cvs

CVS is the source control system used by Mozilla. You do not need cvs if you are only going to build from source tarballs downloaded via FTP. If so, you may skip this step.

To pull Mozilla sources, cvs version 1.11 or higher is required. Cygwin CVS is recommended. WinCVS is also compatible. When installing CVS, you should also install an editor such as emacs or Xemacs.

Compiler & Linker

For doing development on the CVS trunk (Mozilla 1.9 or higher), the standard compiler is Microsoft Visual C++, version 8. The free Express edition of Microsoft Visual C++ 8 (aka Visual C++ 2005 Express Edition) will work with some extra tools and configure options. Visual C++ 7.1 will also work. VC++ 7 is not recommended, and VC6 is not supported and will produce unusable builds.

The build may occur from within either a Windows command line shell or from a cygwin shell; if you use the Windows shell, you need to set up the compiler environment variables using the vcvars32.bat script that is installed along with Visual Studio (selecting "Visual C++ Command Prompt" from the Start menu entries for VS will execute this script for you in a new shell).

It is also possible (but not recommended) to compile Mozilla using the MinGW gcc compiler: see Compiling Mozilla with MinGW.

Table of supported Visual Studio compilers

Branch	HEAD Gecko 1.9 Firefox 3	MOZILLA_1_8_BRANCH Gecko 1.8.1 Firefox 2	MOZILLA_1_8_0_BRANCH Gecko 1.8 Firefox 1.5.0.x
VC6	No	Yes (Official)	Yes (Official)
VC7 (Visual Studio 2002)	Not really	Yes	Yes
VC7.1 (Visual Studio 2003)	Yes	Yes	Yes
VC8 (Visual Studio 2005)	Yes (Official)	No	No
VC9 (Visual Studio Codename "Orcas")	Yes (Experimental)	No	No

Note 1: Firefox 1.5/2.0 / Gecko 1.8.x build instructions are not covered in this document.

Note 2: All information about VC9/"Orcas" relate to the January CTP. It may or may not be correct for previous or future releases.

Note 3: You can build the 1.8 branch with Visual Studio 2005 as follows:
The issue with Visual Studio 2005 is the need to have the .manifest files in the same directory as
the executable files when they are run and the need to update the windows version to at least 0x500 for
Windows RPC stubs

1) Edit the configure file and set WINVER and _WIN32_WINNT to 0x500 (This is for Windows 2000 and above) then run ./configure < your options here >
2) Build as normally (you will eventually get an MSVCRT error for xpt_link)
3) When the error displays:
This copy assumes you are in the mozilla directory
copy xpcom\typelib\xpt\tools*.manifest dist\bin
4) Issue the make command again
5) On build completion, copy the application manifest files from app* into dist\bin.
You can find them by changing directory into the app folder and issuing a dir/s *.manifest

The results of this build have not been thoroughly tested, your mileage may vary.

Platform SDK

Skip this step, if you are using Visual Studio .NET 2003 or later, because you already have all the necessary SDKs for building Mozilla. For older compilers, as well as for the free Visual Studio Express Editions, you need to download SDKs from Microsoft to have the GDI+ headers and library.

You can download the Windows Server 2003 SP1 Platform SDK, or the Windows Server 2003 R2 Platform SDK. You must install the "Windows Core SDK" (Tools, Build Environment, and Redistributable Components) and the "Web Workshop SDK" (Build Environment).

In order for the build environment to pick up this SDK, the SDK must be added to the build environment (see below).

NSIS

Some applications built from the Mozilla codebase now use NSIS for their installer, notably the toolkit apps Firefox, Thunderbird and Sunbird. NSIS is already required during the normal build process because the uninstaller is built with the help of NSIS. If you do not disable building the installer with `ac_add_options --disable-installer` in your `.mozconfig` then install NSIS and ensure its install directory (not the 'Bin' directory in the install directory) is in your PATH. Version 2.17 or greater is recommended.

7-Zip and UPX

If you want to make the highly compressed installer used by Firefox and Thunderbird, you need to install the 7-zip and UPX utilities, make sure they are in your path, and export MOZ_INSTALLER_USE_7ZIP in your environment.

The UPX package is available in Cygwin setup under the Utils category. Do not use the DOS version, as it will not work.

Configure the Environment

The build environment must be configured so that the mozilla build system can find the proper compiler and libraries. This is best done with a DOS batch script. Your build may fail if your paths have spaces or other special characters. Make sure that you did not install software into locations with such paths, except Visual C++.

The following is a batch script (run with cmd.exe or modify the cygwin.bat file) which should be used to configure environment variables:

```
rem --- Set HOME so that cvs and ssh work correctly
rem --- cvs uses HOME to locate your .cvspass file, and ssh to locate your .ssh file
rem --- if you are using ssh, your HOME should match the home directory specified in /etc/passwd. See http
set HOME=C:\home

rem --- Set VCVARS to wherever the MSVC vcvars.bat file is found
set VCVARS=C:\Program Files\Microsoft Visual Studio 8\VC\bin\vcvars32.bat

rem --- Set MSSDK to wherever the MS SDK is installed
rem --- Only required for MSVC7 or the Free MSVC editions that don't come with an SDK
set MSSDK=C:\Program Files\Microsoft Platform SDK

rem --- Set MOZ_TOOLS to wherever you have the moztools packaged installed
set MOZ_TOOLS=C:\moztools

rem --- Set CYGWINBASE to wherever cygwin is installed
rem --- Do not use CYGWIN or else cygserver, cygrunsrv, and Cygwin services will not function properly
rem --- Variable CYGWIN is also used to modify Cygwin's behaviour a little bit.
set CYGWINBASE=C:\cygwin
rem --- Make sure Cygwin does not print out a DOS style path warning
set CYGWIN=nodosfilewarning

rem --- Prepend Cygwin path
rem --- This is necessary so that cygwin find is ahead of windows find.exe in the PATH, but cgywin link is
set PATH=%CYGWINBASE%\bin;%PATH%

rem --- Set MSVC environment vars
call "%VCVARS%"

rem --- Prepend SDK paths
rem --- Only required for MSVC7 or the Free MSVC editions that don't come with an SDK
set PATH=%MSSDK%\bin;%PATH%
set INCLUDE=%MSSDK%\include;%MSSDK%\include\atl;%INCLUDE%
set LIB=%MSSDK%\lib;%LIB%

rem --- moztools comes last
set PATH=%PATH%;%MOZ_TOOLS%\bin

rem --- Now the PATH variable contains:
rem MS-SDK; MSVC; Cygwin; Windows; glib/libIDL; Moztools

rem --- Typically the last thing the script does is launch a cygwin shell
rem --- watch for your ~/.profile and /etc/profile which may overwrite your carefully setup PATH!
bash --login -i
```

For a better terminal, check out PuTTYcyg, a build of the PuTTY SSH client that can serve as a terminal for Cygwin. You can use the same batch file, just install PuTTYcyg somewhere on your path, then replace the last command of the batch file with this one, which starts Cygwin in a PuTTY shell:

```
start putty -cygterm -
```

Fix cygwin path

By default, cygwin puts itself at the beginning of PATH, breaking the order of our carefully constructed PATH. You can either

- edit /etc/profile/ = %CYGWINBASE%\etc\profile (use an editor which preserves Unix line endings - cygwin package `nano`, if in doubt), changing the line

```
PATH=/usr/local/bin:/usr/bin:/bin:/usr/X11R6/bin:$PATH
to
PATH=$PATH:/usr/local/bin:/usr/bin:/bin:/usr/X11R6/bin
```

- or add a .profile or .bash_profile to your home dir.

This is required so that Make selects the right instance of the file "link.exe". If you decide not to mess with the Cygwin PATH, simply rename link.exe to something else. You can also make the batch file do this automatically on startup, and undo it on shutdown. To do that, replace the following line in your batch file:

```
bash --login -i
```

with this:

```
rename link.exe cyglink.exe
bash --login -i
rename cyglink.exe link.exe
```

If you are using this method, don't forget to quit from the Cygwin shell by typing "Exit", instead of just closing the window.

Common Problems, Hints and Restrictions

- Check the Mozilla Build Frequently Asked Questions, they list a number of very often encountered build problems.
- If you have chosen DOS-style line endings when installing cygwin, your source and object trees must be located at a /cygdrive/<c> mount point. It must not be located within your /home.
- Mozilla may not build if you install the moztools directory inside the cygwin directory.
- The build may fail if your PATH environment variable contains quotes ("). Quotes are not properly translated when passed down to the cygwin sub-shells. Quotes are usually not needed so they can be removed.
- Double-check to make sure that you have the complete cygwin package list. If not, launch the cygwin installer and add the missing packages.
  - From the Cygwin Mailing Lists, the behaviour change in make 3.81 was caused by a patch for DOS style paths being removed.

- Debugging Mozilla on Windows FAQ: Tips on how to debug Mozilla on Windows.
- Upgrading a Windows build system from VC6 to VC7?

Tags (2)

Edit tags
- Build documentation
- Developing Mozilla

Attachments (0)

Images 0

No images to display in the gallery.

Attach file

Page last modified 10:24, 2 Aug 2010 by Neil