== How to build expat with cmake (experimental) ==

The cmake based buildsystem for expat works on Windows (cygwin, mingw, Visual

Studio) and should work on all other platform cmake supports.

Assuming ~/expat-2.2.5 is the source directory of expat, add a subdirectory

build and change into that directory:

~/expat-2.2.5$ mkdir build && cd build

~/expat-2.2.5/build$

From that directory, call cmake first, then call make, make test and

make install in the usual way:

~/expat-2.2.5/build$ cmake ..

-- The C compiler identification is GNU

-- The CXX compiler identification is GNU

....

-- Configuring done

-- Generating done

-- Build files have been written to: /home/patrick/expat-2.2.5/build

If you want to specify the install location for your files, append

-DCMAKE\_INSTALL\_PREFIX=/your/install/path to the cmake call.

~/expat-2.2.5/build$ make && make test && make install

Scanning dependencies of target expat

[ 5%] Building C object CMakeFiles/expat.dir/lib/xmlparse.c.o

[ 11%] Building C object CMakeFiles/expat.dir/lib/xmlrole.c.o

....

-- Installing: /usr/local/lib/pkgconfig/expat.pc

-- Installing: /usr/local/bin/xmlwf

-- Installing: /usr/local/share/man/man1/xmlwf.1

For Windows builds, you must make sure to call cmake from an environment where

your compiler is reachable, that means either you call it from the

Visual Studio Command Prompt or when using mingw, you must open a cmd.exe and

make sure that gcc can be called. On Windows, you also might want to specify a

special Generator for CMake:

for Visual Studio builds do:

cmake .. -G "Visual Studio 10" && vcexpress expat.sln

for mingw builds do:

cmake .. -G "MinGW Makefiles" -DCMAKE\_INSTALL\_PREFIX=D:\expat-install

&& gmake && gmake install