NAME

apxs - APache eXtenSion tool

SYNOPSIS

apxs **-g** [**-S** *name=value*] **-n** *modname*

apxs **-q** [**-v**] [**-S** *name=value*] *query* ...

apxs **-c** [**-S** *name=value*] [**-o** *dsofile*] [**-I** *incdir*] [**-D** *name=value*] [**-L** *libdir*] [**-l** *libname*] [**-Wc,***compiler-flags*] [**-Wl,***linker-flags*] *files* ...

apxs **-i** [**-S** *name=value*] [**-n** *modname*] [**-a**] [**-A**] *dso-file* ...

apxs **-e** [**-S** *name=value*] [**-n** *modname*] [**-a**] [**-A**] *dso-file* ...

SUMMARY

apxs is a tool for building and installing extension modules for the Apache HyperText Transfer Protocol (HTTP) server. This is achieved by building a dynamic shared object (DSO) from one or more source or object *files* which then can be loaded into the Apache server under runtime via the LoadModule directive from mod_so.

So to use this extension mechanism your platform has to support the DSO feature and your Apache httpd binary has to be built with the mod_so module. The **apxs** tool automatically complains if this is not the case. You can check this yourself by manually running the command

```
$ httpd -l
```

The module mod_so should be part of the displayed list. If these requirements are fulfilled you can easily extend your Apache server’s functionality by installing your own modules with the DSO

mechanism by the help of this **apxs** tool:

```
$ apxs -i -a -c mod_foo.c
gcc -fpic -DSHARED_MODULE -I/path/to/apache/include -c mod_foo.c
ld -Bshareable -o mod_foo.so mod_foo.o
cp mod_foo.so /path/to/apache/modules/mod_foo.so
chmod 755 /path/to/apache/modules/mod_foo.so
[activating module ‘foo’ in /path/to/apache/etc/httpd.conf]
$ apachectl restart
/path/to/apache/sbin/apachectl restart: httpd not running, trying to start
[Tue Mar 31 11:27:55 1998] [debug] mod_so.c(303): loaded module foo_module
/path/to/apache/sbin/apachectl restart: httpd started
$ _
```

The arguments *files* can be any C source file (.c), a object file (.o) or even a library archive (.a). The **apxs** tool automatically recognizes these extensions and automatically used the C source files for compilation while just using the object and archive files for the linking phase. But when using such pre-compiled objects make sure they are compiled for position independent code (PIC) to be able to use them for a dynamically loaded shared object. For instance with GCC you always just have to use **-fpic**. For other C compilers consult its manual page or at watch for the flags **apxs** uses to compile the object files.

For more details about DSO support in Apache read the documentation of mod_so or perhaps even read the **src/modules/standard/mod_so.c** source file.

OPTIONS

Common Options

-n *modname*

This explicitly sets the module name for the **-i** (install) and **-g** (template generation) option. Use this to explicitly specify the module name. For option **-g** this is required, for option **-i** the **apxs** tool tries to determine the name from the source or (as a fallback) at least by guessing it from the filename.

Query Options

-q Performs a query for variables and environment settings used to build **httpd**. When invoked without *query* parameters, it prints all known variables and their values. The optional **-v** parameter formats the list output. .PP Use this to manually determine settings used to build the **httpd** that will load your module. For instance use INC=-I‘apxs -q INCLUDEDIR‘ .PP inside your own Makefiles if you need manual access to Apache’s C header files.

Configuration Options

-S *name=value*

This option changes the apxs settings described above.

Template Generation Options

-g This generates a subdirectory *name* (see option **-n**) and there two files: A sample module source file named **mod**_*name*.c which can be used as a template for creating your own modules or as a quick start for playing with the apxs mechanism. And a corresponding **Makefile** for even easier build and installing of this module.

DSO Compilation Options

-c This indicates the compilation operation. It first compiles the C source files (.c) of *files* into corresponding object files (.o) and then builds a dynamically shared object in *dsofile* by linking these object files plus the remaining object files (.o and .a) of *files*. If no **-o** option is specified the output file is guessed from the first filename in *files* and thus usually defaults to **mod**_*name*.so.

-o *dsofile*

Explicitly specifies the filename of the created dynamically shared object. If not specified and the name cannot be guessed from the *files* list, the fallback name **mod_unknown.so** is used.

-D *name=value*

This option is directly passed through to the compilation command(s). Use this to add your own defines to the build process.

-I *incdir*

This option is directly passed through to the compilation command(s). Use this to add your own include directories to search to the build process.

-L *libdir*

This option is directly passed through to the linker command. Use this to add your own library

directories to search to the build process.

-l *libname*
This option is directly passed through to the linker command. Use this to add your own libraries to search to the build process.

-Wc,*compiler-flags*
This option passes *compiler-flags* as additional flags to the **libtool --mode=compile** command. Use this to add local compiler-specific options.

-Wl,*linker-flags*
This option passes *linker-flags* as additional flags to the **libtool --mode=link** command. Use this to add local linker-specific options.

-p This option causes apxs to link against the apr/apr-util libraries. This is useful when compiling helper programs that use the apr/apr-util libraries.

DSO Installation and Configuration Options

-i This indicates the installation operation and installs one or more dynamically shared objects into the server's *modules* directory.

-a This activates the module by automatically adding a corresponding LoadModule line to Apache's **httpd.conf** configuration file, or by enabling it if it already exists.

-A Same as option **-a** but the created LoadModule directive is prefixed with a hash sign (#), *i.e.*, the module is just prepared for later activation but initially disabled.

-e This indicates the editing operation, which can be used with the **-a** and **-A** options similarly to the **-i** operation to edit Apache's **httpd.conf** configuration file without attempting to install the module.

EXAMPLES

Assume you have an Apache module named **mod_foo.c** available which should extend Apache's server functionality. To accomplish this you first have to compile the C source into a shared object suitable for loading into the Apache server under runtime via the following command:

```
$ apxs -c mod_foo.c
/path/to/libtool --mode=compile gcc ... -c mod_foo.c
```

```
/path/to/libtool --mode=link gcc ... -o mod_foo.la mod_foo.slo
$ _
```

Then you have to update the Apache configuration by making sure a LoadModule directive is present to load this shared object. To simplify this step **apxs** provides an automatic way to install the shared object in its "modules" directory and updating the **httpd.conf** file accordingly. This can be achieved by running:

```
$ apxs -i -a mod_foo.la
/path/to/instdso.sh mod_foo.la /path/to/apache/modules
/path/to/libtool --mode=install cp mod_foo.la /path/to/apache/modules
...
chmod 755 /path/to/apache/modules/mod_foo.so
[activating module ‘foo’ in /path/to/apache/conf/httpd.conf]
$ _
```

This way a line named

```
LoadModule foo_module modules/mod_foo.so
```

is added to the configuration file if still not present. If you want to have this disabled per default use the **-A** option, *i.e.*

```
$ apxs -i -A mod_foo.c
```

For a quick test of the apxs mechanism you can create a sample Apache module template plus a corresponding Makefile via:

```
$ apxs -g -n foo
Creating [DIR]  foo
Creating [FILE] foo/Makefile
Creating [FILE] foo/modules.mk
Creating [FILE] foo/mod_foo.c
Creating [FILE] foo/.deps
$ _
```

Then you can immediately compile this sample module into a shared object and load it into the Apache server:

```
$ cd foo
$ make all reload
apxs -c mod_foo.c
/path/to/libtool --mode=compile gcc ... -c mod_foo.c
/path/to/libtool --mode=link gcc ... -o mod_foo.la mod_foo.slo
apxs -i -a -n "foo" mod_foo.la
/path/to/instdso.sh mod_foo.la /path/to/apache/modules
/path/to/libtool --mode=install cp mod_foo.la /path/to/apache/modules
...
chmod 755 /path/to/apache/modules/mod_foo.so
[activating module ‘foo’ in /path/to/apache/conf/httpd.conf]
apachectl restart
/path/to/apache/sbin/apachectl restart: httpd not running, trying to start
[Tue Mar 31 11:27:55 1998] [debug] mod_so.c(303): loaded module foo_module
/path/to/apache/sbin/apachectl restart: httpd started
$ _
```