

## NAME

xsubpp - compiler to convert Perl XS code into C code

## SYNOPSIS

```
xsubpp [-v] [-except] [-s pattern] [-prototypes] [-noverversioncheck] [-nolinenumbers] [-noptimize]
[-typemap typemap] [-output filename]... file.xs
```

## DESCRIPTION

This compiler is typically run by the makefiles created by *ExtUtils::MakeMaker* or by *Module::Build* or other Perl module build tools.

*xsubpp* will compile XS code into C code by embedding the constructs necessary to let C functions manipulate Perl values and creates the glue necessary to let Perl access those functions. The compiler uses typemaps to determine how to map C function parameters and variables to Perl values.

The compiler will search for typemap files called *typemap*. It will use the following search path to find default typemaps, with the rightmost typemap taking precedence.

```
../../../../typemap:../../../../typemap:../typemap:typemap
```

It will also use a default typemap installed as `ExtUtils::typemap`.

## OPTIONS

Note that the `XSOPT MakeMaker` option may be used to add these options to any makefiles generated by *MakeMaker*.

### **-hiertype**

Retains ':' in type names so that C++ hierarchical types can be mapped.

### **-except**

Adds exception handling stubs to the C code.

### **-typemap typemap**

Indicates that a user-supplied typemap should take precedence over the default typemaps. This option may be used multiple times, with the last typemap having the highest precedence.

### **-output filename**

Specifies the name of the output file to generate. If no file is specified, output will be written to standard output.

### **-v**

Prints the *xsubpp* version number to standard output, then exits.

### **-prototypes**

By default *xsubpp* will not automatically generate prototype code for all xsubs. This flag will enable prototypes.

### **-noverversioncheck**

Disables the run time test that determines if the object file (derived from the `.xs` file) and the `.pm` files have the same version number.

### **-nolinenumbers**

Prevents the inclusion of '#line' directives in the output.

### **-noptimize**

Disables certain optimizations. The only optimization that is currently affected is the use of *targets* by the output C code (see *perlguts*). This may significantly slow down the generated code, but this is the way **xsubpp** of 5.005 and earlier operated.

**-noinout**

Disable recognition of `IN`, `OUT_LIST` and `INOUT_LIST` declarations.

**-noargtypes**

Disable recognition of ANSI-like descriptions of function signature.

**-C++**

Currently doesn't do anything at all. This flag has been a no-op for many versions of perl, at least as far back as perl5.003\_07. It's allowed here for backwards compatibility.

**-s=... or -strip=...**

*This option is obscure and discouraged.*

If specified, the given string will be stripped off from the beginning of the C function name in the generated XS functions (if it starts with that prefix). This only applies to XSUBs without `CODE` or `PPCODE` blocks. For example, the XS:

```
void foo_bar(int i);
```

when `xsubpp` is invoked with `-s foo_` will install a `foo_bar` function in Perl, but really call `bar(i)` in C. Most of the time, this is the opposite of what you want and failure modes are somewhat obscure, so please avoid this option where possible.

**ENVIRONMENT**

No environment variables are used.

**AUTHOR**

Originally by Larry Wall. Turned into the `ExtUtils::ParseXS` module by Ken Williams.

**MODIFICATION HISTORY**

See the file *Changes*.

**SEE ALSO**

`perl(1)`, `perlxs(1)`, `perlxsut(1)`, `ExtUtils::ParseXS`