

NAME

ExtUtils::Typemaps - Read/Write/Modify Perl/XS typemap files

SYNOPSIS

```
# read/create file
my $typemap = ExtUtils::Typemaps->new(file => 'typemap');
# alternatively create an in-memory typemap
# $typemap = ExtUtils::Typemaps->new();
# alternatively create an in-memory typemap by parsing a string
# $typemap = ExtUtils::Typemaps->new(string => $sometypemap);

# add a mapping
$typemap->add_typemap(ctype => 'NV', xstype => 'T_NV');
$typemap->add_inputmap(
  xstype => 'T_NV', code => '$var = ($type)SvNV($arg);'
);
$typemap->add_outputmap(
  xstype => 'T_NV', code => 'sv_setnv($arg, (NV)$var);'
);
$typemap->add_string(string => $typemapstring);
# will be parsed and merged

# remove a mapping (same for remove_typemap and remove_outputmap...)
$typemap->remove_inputmap(xstype => 'SomeType');

# save a typemap to a file
$typemap->write(file => 'anotherfile.map');

# merge the other typemap into this one
$typemap->merge(typemap => $another_typemap);
```

DESCRIPTION

This module can read, modify, create and write Perl XS typemap files. If you don't know what a typemap is, please confer the [perlxstut\(1\)](#) and [perlxs\(1\)](#) manuals.

The module is not entirely round-trip safe: For example it currently simply strips all comments. The order of entries in the maps is, however, preserved.

We check for duplicate entries in the typemap, but do not check for missing TYPEMAP entries for INPUTMAP or OUTPUTMAP entries since these might be hidden in a different typemap.

METHODS**new**

Returns a new typemap object. Takes an optional `file` parameter. If set, the given file will be read. If the file doesn't exist, an empty typemap is returned.

Alternatively, if the `string` parameter is given, the supplied string will be parsed instead of a file.

file

Get/set the file that the typemap is written to when the `write` method is called.

add_typemap

Add a TYPEMAP entry to the typemap.

Required named arguments: The `ctype` (e.g. `ctype => 'double'`) and the `xstype` (e.g. `xstype => 'T_NV'`).

Optional named arguments: `replace => 1` forces removal/replacement of existing TYPEMAP entries of the same `ctype`. `skip => 1` triggers a “*first come first serve*” logic by which new entries that conflict with existing entries are silently ignored.

As an alternative to the named parameters usage, you may pass in an [ExtUtils::Typemaps::Type](#) object as first argument, a copy of which will be added to the typemap. In that case, only the `replace` or `skip` named parameters may be used after the object. Example:

```
$map->add_typemap($type_obj, replace => 1);
```

add_inputmap

Add an INPUT entry to the typemap.

Required named arguments: The `xstype` (e.g. `xstype => 'T_NV'`) and the code to associate with it for input.

Optional named arguments: `replace => 1` forces removal/replacement of existing INPUT entries of the same `xstype`. `skip => 1` triggers a “*first come first serve*” logic by which new entries that conflict with existing entries are silently ignored.

As an alternative to the named parameters usage, you may pass in an [ExtUtils::Typemaps::InputMap](#) object as first argument, a copy of which will be added to the typemap. In that case, only the `replace` or `skip` named parameters may be used after the object. Example:

```
$map->add_inputmap($type_obj, replace => 1);
```

add_outputmap

Add an OUTPUT entry to the typemap. Works exactly the same as `add_inputmap`.

add_string

Parses a string as a typemap and merge it into the typemap object.

Required named argument: `string` to specify the string to parse.

remove_typemap

Removes a TYPEMAP entry from the typemap.

Required named argument: `ctype` to specify the entry to remove from the typemap.

Alternatively, you may pass a single [ExtUtils::Typemaps::Type](#) object.

remove_inputmap

Removes an INPUT entry from the typemap.

Required named argument: `xstype` to specify the entry to remove from the typemap.

Alternatively, you may pass a single [ExtUtils::Typemaps::InputMap](#) object.

remove_outputmap

Removes an OUTPUT entry from the typemap.

Required named argument: `xstype` to specify the entry to remove from the typemap.

Alternatively, you may pass a single [ExtUtils::Typemaps::OutputMap](#) object.

get_typemap

Fetches an entry of the TYPEMAP section of the typemap.

Mandatory named arguments: The `ctype` of the entry.

Returns the [ExtUtils::Typemaps::Type](#) object for the entry if found.

get_inputmap

Fetches an entry of the INPUT section of the typemap.

Mandatory named arguments: The `xstype` of the entry or the `ctype` of the typemap that can be used to find the `xstype`. To wit, the following pieces of code are equivalent:

```
my $type = $typemap->get_typemap(ctype => $ctype)
my $input_map = $typemap->get_inputmap(xstype => $type->xstype);
```

```
my $input_map = $typemap->get_inputmap(ctype => $ctype);
```

Returns the `ExtUtils::Typemaps::InputMap` object for the entry if found.

get_outputmap

Fetches an entry of the OUTPUT section of the typemap.

Mandatory named arguments: The `xstype` of the entry or the `ctype` of the typemap that can be used to resolve the `xstype`. (See above for an example.)

Returns the `ExtUtils::Typemaps::InputMap` object for the entry if found.

write

Write the typemap to a file. Optionally takes a `file` argument. If given, the typemap will be written to the specified file. If not, the typemap is written to the currently stored file name (see `->file` above, this defaults to the file it was read from if any).

as_string

Generates and returns the string form of the typemap.

as_embedded_typemap

Generates and returns the string form of the typemap with the appropriate prefix around it for verbatim inclusion into an XS file as an embedded typemap. This will return a string like

```
TYPEMAP: <<END_OF_TYPEMAP
... typemap here (see as_string) ...
END_OF_TYPEMAP
```

The method takes care not to use a HERE-doc end marker that appears in the typemap string itself.

merge

Merges a given typemap into the object. Note that a failed merge operation leaves the object in an inconsistent state so clone it if necessary.

Mandatory named arguments: Either `typemap => $another_typemap_obj` or `file => $path_to_typemap_file` but not both.

Optional arguments: `replace => 1` to force replacement of existing typemap entries without warning or `skip => 1` to skip entries that exist already in the typemap.

is_empty

Returns a bool indicating whether this typemap is entirely empty.

list_mapped_ctypes

Returns a list of the C types that are mappable by this typemap object.

_get_typemap_hash

Returns a hash mapping the C types to the XS types:

```
{
  'char **' => 'T_PACKEDARRAY',
  'bool_t' => 'T_IV',
  'AV *' => 'T_AVREF',
  'InputStream' => 'T_IN',
  'double' => 'T_DOUBLE',
  # ...
}
```

This is documented because it is used by `ExtUtils::ParseXS` but it's not intended for general consumption. May be removed at any time.

_get_inputmap_hash

Returns a hash mapping the XS types (identifiers) to the corresponding INPUT code:

```

{
'T_CALLBACK' => ' $var = make_perl_cb_{$type}($arg)
',
'T_OUT' => ' $var = IoOFP(sv_2io($arg))
',
'T_REF_IV_PTR' => ' if (sv_isa($arg, "\\${ntype}\\")) {
# ...
}

```

This is documented because it is used by [ExtUtils::ParseXS](#) but it's not intended for general consumption. May be removed at any time.

get_outputmap_hash

Returns a hash mapping the XS types (identifiers) to the corresponding OUTPUT code:

```

{
'T_CALLBACK' => ' sv_setpvn($arg, $var.context.value().chp(),
$var.context.value().size());
',
'T_OUT' => ' {
GV *gv = newGVgen("$Package");
if ( do_open(gv, "+>&", 3, FALSE, 0, 0, $var) )
sv_setsv(
$arg,
sv_bless(newRV((SV*)gv), gv_stashpv("$Package",1))
);
else
$arg = &PL_sv_undef;
}
',
# ...
}

```

This is documented because it is used by [ExtUtils::ParseXS](#) but it's not intended for general consumption. May be removed at any time.

get_prototype_hash

Returns a hash mapping the C types of the typemap to their corresponding prototypes.

```

{
'char **' => '$',
'bool_t' => '$',
'AV *' => '$',
'InputStream' => '$',
'double' => '$',
# ...
}

```

This is documented because it is used by [ExtUtils::ParseXS](#) but it's not intended for general consumption. May be removed at any time.

clone

Creates and returns a clone of a full typemaps object.

Takes named parameters: If `shallow` is true, the clone will share the actual individual type/input/outputmap objects, but not share their storage. Use with caution. Without `shallow`, the clone will be fully independent.

tidy_type

Function to (heuristically) canonicalize a C type. Works to some degree with C++ types.

```
$halfway_canonical_type = tidy_type($ctype);
```

Moved from [ExtUtils::ParseXS](#)

CAVEATS

Inherits some evil code from [ExtUtils::ParseXS](#)

SEE ALSO

The parser is heavily inspired from the one in [ExtUtils::ParseXS](#).

For details on typemaps: [perlxstut\(1\)](#), [perlxs](#).

AUTHOR

Steffen Mueller <smueller@cpan.org>

COPYRIGHT & LICENSE

Copyright 2009, 2010, 2011, 2012, 2013 Steffen Mueller

This program is free software; you can redistribute it and/or modify it under the same terms as Perl itself.