

## NAME

if - use a Perl module if a condition holds (also can `no` a module)

## SYNOPSIS

```
use if CONDITION, MODULE => ARGUMENTS;
no if CONDITION, MODULE => ARGUMENTS;
```

## DESCRIPTION

The `if` module is used to conditionally load or unload another module. The construct

```
use if CONDITION, MODULE => ARGUMENTS;
```

will load `MODULE` only if `CONDITION` evaluates to true. The above statement has no effect unless `CONDITION` is true. If the `CONDITION` does evaluate to true, then the above line has the same effect as:

```
use MODULE ARGUMENTS;
```

The use of `=>` above provides necessary quoting of `MODULE`. If you don't use the fat comma (eg you don't have any `ARGUMENTS`), then you'll need to quote the `MODULE`.

## EXAMPLES

The following line is taken from the testsuite for *File::Map*:

```
use if $^O ne 'MSWin32', POSIX => qw/setlocale LC_ALL/;
```

If run on any operating system other than Windows, this will import the functions `setlocale` and `LC_ALL` from *POSIX*. On Windows it does nothing.

The following is used to *deprecate* core modules beyond a certain version of Perl:

```
use if $] > 5.016, 'deprecate';
```

This line is taken from *Text::Soundex* 3.04, and marks it as deprecated beyond Perl 5.16. If you `use Text::Soundex` in Perl 5.18, for example, and you have used *warnings*, then you'll get a warning message (the `deprecate` module looks to see whether the calling module was `use`'d from a core library directory, and if so, generates a warning), unless you've installed a more recent version of *Text::Soundex* from CPAN.

You can also specify to NOT use something:

```
no if $] ge 5.021_006, warnings => "locale";
```

This warning category was added in the specified Perl version (a development release). Without the `'if'`, trying to use it in an earlier release would generate an unknown warning category error.

## BUGS

The current implementation does not allow specification of the required version of the module.

## SEE ALSO

*Module::Requires* can be used to conditionally load one or modules, with constraints based on the version of the module. Unlike `if` though, *Module::Requires* is not a core module.

*Module::Load::Conditional* provides a number of functions you can use to query what modules are available, and then load one or more of them at runtime.

---

*provide* can be used to select one of several possible modules to load, based on what version of Perl is running.

## AUTHOR

Ilya Zakharevich *mailto:ilyaz@cpan.org*.

## COPYRIGHT AND LICENCE

This software is copyright (c) 2002 by Ilya Zakharevich.

This is free software; you can redistribute it and/or modify it under the same terms as the Perl 5 programming language system itself.