NAME

MRO::Compat - mro::* interface compatibility for Perls < 5.9.5

SYNOPSIS

```
package PPP; use base qw/Exporter/;
package X; use base qw/PPP/;
package Y; use base qw/PPP/;
package Z; use base qw/PPP/;

package FooClass; use base qw/X Y Z/;

package main;
use MRO::Compat;
my $linear = mro::get_linear_isa('FooClass');
print join(q{, }, @$linear);

# Prints: FooClass, X, PPP, Exporter, Y, Z
```

DESCRIPTION

The "mro" namespace provides several utilities for dealing with method resolution order and method caching in general in Perl 5.9.5 and higher.

This module provides those interfaces for earlier versions of Perl (back to 5.6.0 anyways).

It is a harmless no-op to use this module on 5.9.5+. That is to say, code which properly uses MRO::Compat will work unmodified on both older Perls and 5.9.5+.

If you're writing a piece of software that would like to use the parts of 5.9.5+'s mro:: interfaces that are supported here, and you want compatibility with older Perls, this is the module for you.

Some parts of this code will work better and/or faster with Class::C3::XS installed (which is an optional prereq of Class::C3, which is in turn a prereq of this package), but it's not a requirement.

This module never exports any functions. All calls must be fully qualified with the `mro::` prefix.

The interface documentation here serves only as a quick reference of what the function basically does, and what differences between MRO::Compat and 5.9.5+ one should look out for. The main docs in 5.9.5's mro are the real interface docs, and contain a lot of other useful information.

Functions

mro::get_linear_isa($classname[, $type])

Returns an arrayref which is the linearized "ISA" of the given class. Uses whichever MRO is currently in effect for that class by default, or the given MRO (either `c3` or `dfs` if specified as `$type`).

The linearized ISA of a class is a single ordered list of all of the classes that would be visited in the process of resolving a method on the given class, starting with itself. It does not include any duplicate entries.

Note that `UNIVERSAL` (and any members of `UNIVERSAL`'s MRO) are not part of the MRO of a class, even though all classes implicitly inherit methods from `UNIVERSAL` and its parents.

mro::import

This allows the `use mro 'dfs'` and `use mro 'c3'` syntaxes, providing you "use MRO::Compat" first. Please see the "USING C3" section for additional details.

mro::set_mro($classname, $type)

Sets the mro of `$classname` to one of the types `dfs` or `c3`. Please see the "USING C3" section for additional details.

mro::get_mro($classname)

Returns the MRO of the given class (either `c3` or `dfs`).

It considers any Class::C3-using class to have C3 MRO even before *Class::C3::initialize()* is called.

mro::get_isarev($classname)
Returns an arrayref of classes who are subclasses of the given classname. In other words, classes in whose `@ISA` hierarchy we appear, no matter how indirectly.

This is much slower on pre-5.9.5 Perls with MRO::Compat than it is on 5.9.5+, as it has to search the entire package namespace.

mro::is_universal($classname)
Returns a boolean status indicating whether or not the given classname is either `UNIVERSAL` itself, or one of `UNIVERSAL`'s parents by `@ISA` inheritance.

Any class for which this function returns true is "universal" in the sense that all classes potentially inherit methods from it.

mro::invalidate_all_method_caches
Increments `PL_sub_generation`, which invalidates method caching in all packages.

Please note that this is rarely necessary, unless you are dealing with a situation which is known to confuse Perl's method caching.

mro::method_changed_in($classname)
Invalidates the method cache of any classes dependent on the given class. In MRO::Compat on pre-5.9.5 Perls, this is an alias for `mro::invalidate_all_method_caches` above, as pre-5.9.5 Perls have no other way to do this. It will still enforce the requirement that you pass it a classname, for compatibility.

Please note that this is rarely necessary, unless you are dealing with a situation which is known to confuse Perl's method caching.

mro::get_pkg_gen($classname)
Returns an integer which is incremented every time a local method of or the `@ISA` of the given package changes on Perl 5.9.5+. On earlier Perls with this MRO::Compat module, it will probably increment a lot more often than necessary.

USING C3

While this module makes the 5.9.5+ syntaxes `use mro 'c3'` and `mro::set_mro("Foo", 'c3')` available on older Perls, it does so merely by passing off the work to Class::C3.

It does not remove the need for you to call `Class::C3::initialize()`, `Class::C3::reinitialize()`, and/or `Class::C3::uninitialize()` at the appropriate times as documented in the Class::C3 docs. These three functions are always provided by MRO::Compat, either via Class::C3 itself on older Perls, or directly as no-ops on 5.9.5+.

SEE ALSO

Class::C3

mro

AUTHOR

Brandon L. Black, <blblack@gmail.com>

COPYRIGHT AND LICENSE