NAME

parent - Establish an ISA relationship with base classes at compile time

SYNOPSIS

```
package Baz;
use parent qw(Foo Bar);
```

DESCRIPTION

Allows you to both load one or more modules, while setting up inheritance from those modules at the same time. Mostly similar in effect to

```
package Baz;
BEGIN {
require Foo;
require Bar;
push @ISA, qw(Foo Bar);
}
```

By default, every base class needs to live in a file of its own. If you want to have a subclass and its parent class in the same file, you can tell `parent` not to load any modules by using the `-norequire` switch:

```
package Foo;
sub exclaim { "I CAN HAS PERL" }

package DoesNotLoadFooBar;
use parent -norequire, 'Foo', 'Bar';
# will not go looking for Foo.pm or Bar.pm
```

This is equivalent to the following code:

```
package Foo;
sub exclaim { "I CAN HAS PERL" }

package DoesNotLoadFooBar;
push @DoesNotLoadFooBar::ISA, 'Foo', 'Bar';
```

This is also helpful for the case where a package lives within a differently named file:

```
package MyHash;
use Tie::Hash;
use parent -norequire, 'Tie::StdHash';
```

This is equivalent to the following code:

```
package MyHash;
require Tie::Hash;
push @ISA, 'Tie::StdHash';
```

If you want to load a subclass from a file that `require` would not consider an eligible filename (that is, it does not end in either `.pm` or `.pmc`), use the following code:

```
package MySecondPlugin;
require './plugins/custom.plugin'; # contains Plugin::Custom
use parent -norequire, 'Plugin::Custom';
```

HISTORY

This module was forked from base to remove the cruft that had accumulated in it.

CAVEATS

SEE ALSO

base

AUTHORS AND CONTRIBUTORS

Rafaël Garcia-Suarez, Bart Lateur, Max Maischein, Anno Siegel, Michael Schwern

MAINTAINER

Max Maischein `corion@cpan.org`

LICENSE