## NAME

UNIVERSAL - base class for ALL classes (blessed references)

## SYNOPSIS

```
    $is_io = $fd->isa("IO::Handle");
    $is_io = Class->isa("IO::Handle");

    $sub = $obj->can("print");
    $sub = Class->can("print");

    use UNIVERSAL qw( isa can VERSION );
    $yes = isa $ref, "HASH" ;
    $sub = can $ref, "fandango" ;
    $ver = VERSION $obj ;
```

## DESCRIPTION

`UNIVERSAL` is the base class which all bless references will inherit from, see *perlobj*.

`UNIVERSAL` provides the following methods and functions:

`$obj->isa( TYPE )`

`CLASS->isa( TYPE )`

`isa( VAL, TYPE )`

Where

`TYPE`

is a package name

`$obj`

is a blessed reference or a string containing a package name

`CLASS`

is a package name

`VAL`

is any of the above or an unblessed reference

When used as an instance or class method (`$obj->isa( TYPE )`), `isa` returns *true* if $obj is blessed into package `TYPE` or inherits from package `TYPE`.

When used as a class method (`CLASS->isa( TYPE )`: sometimes referred to as a static method), `isa` returns *true* if `CLASS` inherits from (or is itself) the name of the package `TYPE` or inherits from package `TYPE`.

When used as a function, like

```
   use UNIVERSAL qw( isa ) ;
   $yes = isa $h, "HASH";
   $yes = isa "Foo", "Bar";
```

or

```
   require UNIVERSAL ;
   $yes = UNIVERSAL::isa $a, "ARRAY";
```

`isa` returns *true* in the same cases as above and also if `VAL` is an unblessed reference to a perl variable of type `TYPE`, such as "HASH", "ARRAY", or "Regexp".

```
$obj->can( METHOD )
CLASS->can( METHOD )
can( VAL, METHOD )
```

can checks if the object or class has a method called METHOD. If it does then a reference to the sub is returned. If it does not then *undef* is returned. This includes methods inherited or imported by $obj, CLASS, or VAL.

can cannot know whether an object will be able to provide a method through AUTOLOAD, so a return value of *undef* does not necessarily mean the object will not be able to handle the method call. To get around this some module authors use a forward declaration (see *perlsub*) for methods they will handle via AUTOLOAD. For such 'dummy' subs, can will still return a code reference, which, when called, will fall through to the AUTOLOAD. If no suitable AUTOLOAD is provided, calling the coderef will cause an error.

can can be called as a class (static) method, an object method, or a function.

When used as a function, if VAL is a blessed reference or package name which has a method called METHOD, can returns a reference to the subroutine. If VAL is not a blessed reference, or if it does not have a method METHOD, *undef* is returned.

```
VERSION ( [ REQUIRE ] )
```

VERSION will return the value of the variable $VERSION in the package the object is blessed into. If REQUIRE is given then it will do a comparison and die if the package version is not greater than or equal to REQUIRE.

VERSION can be called as either a class (static) method, an object method or a function.

## EXPORTS

None by default.

You may request the import of all three functions (isa, can, and VERSION), however it isn't usually necessary to do so. Perl magically makes these functions act as methods on all objects. The one exception is isa, which is useful as a function when operating on non-blessed references.