NAME

arybase - Set indexing base via $[

SYNOPSIS

```
$[ = 1;

@a = qw(Sun Mon Tue Wed Thu Fri Sat);
print $a[3], "\n"; # prints Tue
```

DESCRIPTION

This module implements Perl's `$[` variable. You should not use it directly.

Assigning to `$[` has the *compile-time* effect of making the assigned value, converted to an integer, the index of the first element in an array and the first character in a substring, within the enclosing lexical scope.

It can be written with or without `local`:

```
$[ = 1;
local $[ = 1;
```

It only works if the assignment can be detected at compile time and the value assigned is constant.

It affects the following operations:

```
$array[$element]
@array[@slice]
$#array
(list())[$slice]
splice @array, $index, ...
each @array
keys @array

index $string, $substring # return value is affected
pos $string
substr $string, $offset, ...
```

As with the default base of 0, negative bases count from the end of the array or string, starting with -1. If `$[` is a positive integer, indices from `$[-1` to 0 also count from the end. If `$[` is negative (why would you do that, though?), indices from `$[` to 0 count from the beginning of the string, but indices below `$[` count from the end of the string as though the base were 0.

Prior to Perl 5.16, indices from 0 to `$[-1` inclusive, for positive values of `$[`, behaved differently for different operations; negative indices equal to or greater than a negative `$[` likewise behaved inconsistently.

HISTORY

Before Perl 5, `$[` was a global variable that affected all array indices and string offsets.

Starting with Perl 5, it became a file-scoped compile-time directive, which could be made lexically-scoped with `local`. ''File-scoped'' means that the `$[` assignment could leak out of the block in which occurred:

```
{
$[ = 1;
# ... array base is 1 here ...
}
# ... still 1, but not in other files ...
```

In Perl 5.10, it became strictly lexical. The file-scoped behaviour was removed (perhaps inadvertently, but what's done is done).

In Perl 5.16, the implementation was moved into this module, and out of the Perl core. The erratic behaviour that occurred with indices between -1 and `$[` was made consistent between operations, and, for negative bases, indices from `$[` to -1 inclusive were made consistent between operations.

BUGS

Error messages that mention array indices use the 0-based index.

`keys $arrayref`and `each $arrayref` do not respect the curren t value of `$[`.

SEE ALSO

"$[" in perlvar(1), Array::Base and String::Base.