NAME

Exporter::Declare::Specs − Import argument parser for Exporter::Declare

DESCRIPTION

Import arguments can get complicated. All arguments are assumed to be exports unless they have a − or : prefix. The prefix may denote a tag, a boolean option, or an option that takes the next argument as a value. In addition almost all these can be negated with the ! prefix.

This class takes care of parsing the import arguments and generating data structures that can be used to find what the exporter needs to know.

METHODS

`$class`−>new(`$package`, `@args`)

Create a new instance and parse `@args`.

`$specs`−>*package()*

Get the name of the package that should do the exporting.

`$hashref` = `$specs`−>*config()*

Get the configuration hash, All specified options and tags are the keys. The value will be true/false/undef for tags/boolean options. For options that take arguments the value will be that argument. When a config hash is provided to a tag it will be the value.

`@names` = `$specs`−>*arguments()*
`@names` = `$specs`−>*options()*
`@names` = `$specs`−>*tags()*

Get the argument, option, or tag names that were specified for the import.

`$hashref` = `$specs`−>*argument_info()*

Get the arguments that were specified for the import. The key is the name of the argument and the value is what the user supplied during import.

`$hashref` = `$specs`−>*option_info()*

Get the options that were specified for the import. The key is the name of the user supplied option and the value will evaluate to true.

`$hashref` = `$specs`−>*tag_info()*

Get the values associated with the tags used during import. The key is the name of the tag and the value is an array ref containing the values given to *export_tag()* for the associated name.

`$hashref` = `$specs`−>*exports()*

Get the exports hash. The keys are names of the exports. Values are an array containing the export, item specific config hash, and arguments array. This is generally not intended for direct consumption.

`$arrayref` = `$specs`−>*excludes()*

Get the arrayref containing the names of all excluded exports.

`$specs`−>export(`$package`)

Do the actual exporting. All exports will be injected into `$package`.

`$specs`−>add_export(`$name`, `$value`)
`$specs`−>add_export(`$name`, `$value`, \%config)

Add an export. Name is required, including sigil. Value is required, if it is a sub it will be blessed as a ::Sub, otherwise blessed as a ::Variable.

```
$specs−>add_export( '&foo' => sub { return 'foo' });
```

AUTHORS

Chad Granum exodist7 AT gmail DOT com

COPYRIGHT