NAME

Pod::Simple::PullParserToken -- tokens from Pod::Simple::PullParser

SYNOPSIS

Given a $parser that's an object of class Pod::Simple::PullParser (or a subclass)...

```
  while(my $token = $parser->get_token) {
    $DEBUG and print "Token: ", $token->dump, "\n";
    if($token->is_start) {
      ...access $token->tagname, $token->attr, etc...

    } elsif($token->is_text) {
      ...access $token->text, $token->text_r, etc...

    } elsif($token->is_end) {
      ...access $token->tagname...

    }
  }
```

(Also see *Pod::Simple::PullParser*)

DESCRIPTION

When you do $parser->get_token on a *Pod::Simple::PullParser*, you should get an object of a subclass of Pod::Simple::PullParserToken.

Subclasses will add methods, and will also inherit these methods:

$token->type

> This returns the type of the token. This will be either the string "start", the string "text", or the string "end".
>
> Once you know what the type of an object is, you then know what subclass it belongs to, and therefore what methods it supports.
>
> Yes, you could probably do the same thing with code like $token->isa('Pod::Simple::PullParserEndToken'), but that's not so pretty as using just $token->type, or even the following shortcuts:

$token->is_start

> This is a shortcut for `$token->type() eq "start"`

$token->is_text

> This is a shortcut for `$token->type() eq "text"`

$token->is_end

> This is a shortcut for `$token->type() eq "end"`

$token->dump

> This returns a handy stringified value of this object. This is useful for debugging, as in:
>
> ```
>   while(my $token = $parser->get_token) {
>     $DEBUG and print "Token: ", $token->dump, "\n";
>     ...
>   }
> ```

SEE ALSO

My subclasses: *Pod::Simple::PullParserStartToken*, *Pod::Simple::PullParserTextToken*, and *Pod::Simple::PullParserEndToken*.

Pod::Simple::PullParser and *Pod::Simple*

COPYRIGHT AND DISCLAIMERS

AUTHOR

Sean M. Burke `sburke@cpan.org`