POD Translation by *pod2pdf*

ajf@afco.demon.co.uk

Field.pm

Table of Contents
Field.pm

NAME

MARC::Field - Perl extension for handling MARC fields

VERSION

Version 1.13

```
$Id: Field.pdf,v 1.1 2002/12/02 21:40:58 edsummers Exp $
```

SYNOPSIS

```
use MARC::Field;

my $field =
      MARC::Field->new(
              245, '1', '0',
                      'a' => 'Raccoons and ripe corn / ',
                      'c' => 'Jim Arnosky.'
              );
$field->add_subfields( "a", "1st ed." );
```

DESCRIPTION

Defines MARC fields for use in the MARC::Record module. I suppose you could use them on their own, but that wouldn't be very interesting.

EXPORT

None by default. Any errors are stored in `$MARC::Field::ERROR`, which `$MARC::Record` usually bubbles up to `$MARC::Record::ERROR`.

METHODS

new(tag,indicator1,indicator2,code,data[,code,data...])

```
my $record =
      MARC::Field->new(
              245, '1', '0',
                      'a' => 'Raccoons and ripe corn / ',
                      'c' => 'Jim Arnosky.'
              );
```

Returns a MARC::Field record.

clone()

Makes a copy of the field. Note that this is not just the same as saying

```
my $newfield = $field;
```

since that just makes a copy of the reference. To get a new object, you must

```
my $newfield = $field->clone;
```

Returns a MARC::Field record.

update()

Allows you to change the values of the field. You can update indicators and subfields like this:

```
$field->update( ind2 => '4', a => 'The ballad of Abe Lincoln');
```

If you want to update a field that has no indicators or subfields (000-009) just call update() with one argument, the string that you would like to set the field to.

```
$field = $record->field( '003' );
$field->update('IMchF');
```

Note: when doing subfield updates be aware that `update()` will only update the first occurrence. If you need to do anything more complicated you need to create a new field and use `replace_with()`. Returns the number of items modified.

replace_with()

Allows you to replace an existing field with a new one. You need to pass `replace()` a MARC::Field object to replace the existing field with. For example:

```
$field = $record->field('245');
my $new_field = new MARC::Field('245','0','4','The ballad of Abe Lincoln.');
$field->replace_with($new_field);
```

Doesn't return a meaningful or reliable value.

tag()

Returns the three digit tag for the field.

indicator(indno)

Returns the specified indicator. Returns `undef` and sets `$MARC::Field::ERROR` if the *indno* is not 1 or 2, or if the tag doesn't have indicators.

subfield(code)

Returns the text from the first subfield matching the subfield code. If no matching subfields are found, `undef` is returned.

If the tag is less than an 010, `undef` is returned and `$MARC::Field::ERROR` is set.

```
my $subA = $field->subfield('a');
```

subfields()

Returns all the subfields in the field. What's returned is a list of lists, where the inner list is a subfield code and the subfield data.

For example, this might be the subfields from a 245 field:

```
[
  [ 'a', 'Perl in a nutshell :' ],
  [ 'b', 'A desktop quick reference.' ],
]
```

data()

Returns the data part of the field, if the tag number is less than 10.

add_subfields(code,text[,code,text ...])

Adds subfields to the end of the subfield list.

Returns the number of subfields added, or `undef` if there was an error.

as_string()

Returns a string of all subfields run together, without the tag number.

as_formatted()

Returns a pretty string for printing in a MARC dump.

as_usmarc()

Returns a string for putting into a USMARC file. It's really only useful by `MARC::Record::as_usmarc()`.

warnings()

Returns the warnings that were created when the record was read. These are things like "Invalid indicators converted to blanks".

The warnings are items that you might be interested in, or might not. It depends on how stringently you're checking data. If you're doing some grunt data analysis, you probably don't care.

SEE ALSO

See the "SEE ALSO" section for *MARC::Record*.

TODO

See the "TODO" section for *MARC::Record*.

LICENSE

This code may be distributed under the same terms as Perl itself.

Please note that these modules are not products of or supported by the employers of the various contributors to the code.

AUTHOR

Andy Lester, <marc@petdance.com> or <alester@flr.follett.com>