

**NAME**

Net::DNS::Text – DNS text representation

**SYNOPSIS**

```
use Net::DNS::Text;

$object = new Net::DNS::Text('example');
$string = $object->string;

$object = decode Net::DNS::Text( \ $data, $offset );
( $object, $next ) = decode Net::DNS::Text( \ $data, $offset );

$data = $object->encode;
$text = $object->value;
```

**DESCRIPTION**

The `Net::DNS::Text` module implements a class of text objects with associated class and instance methods.

Each text object instance has a fixed identity throughout its lifetime.

**METHODS****new**

```
$object = new Net::DNS::Text('example');
```

Creates a text object which encapsulates a single character string component of a resource record.

Arbitrary single-byte characters can be represented by `\` followed by exactly three decimal digits. Such characters are devoid of any special meaning.

A character preceded by `\` represents itself, without any special interpretation.

**decode**

```
$object = decode Net::DNS::Text( \ $buffer, $offset );

( $object, $next ) = decode Net::DNS::Text( \ $buffer, $offset );
```

Creates a text object which represents the decoded data at the indicated offset within the data buffer.

The argument list consists of a reference to a scalar containing the wire-format data and offset of the text data.

The returned offset value indicates the start of the next item in the data buffer.

**encode**

```
$data = $object->encode;
```

Returns the wire-format encoded representation of the text object suitable for inclusion in a DNS packet buffer.

**raw**

```
$data = $object->raw;
```

Returns the wire-format encoded representation of the text object without the explicit length field.

**value**

```
$value = $text->value;
```

Character string representation of the text object.

**string**

```
$string = $text->string;
```

Conditionally quoted zone file representation of the text object.

**BUGS**

Coding strategy is intended to avoid creating unnecessary argument lists and stack frames. This improves efficiency at the expense of code readability.

Platform specific character coding features are conditionally compiled into the code.

**COPYRIGHT**

Copyright (c)2009–2011 Dick Franks.

All rights reserved.

**LICENSE**

Permission to use, copy, modify, and distribute this software and its documentation for any purpose and without fee is hereby granted, provided that the above copyright notice appear in all copies and that both that copyright notice and this permission notice appear in supporting documentation, and that the name of the author not be used in advertising or publicity pertaining to distribution of the software without specific prior written permission.

THE SOFTWARE IS PROVIDED “AS IS”, WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.

**SEE ALSO**

perl, Net::DNS, RFC1035, RFC3629, Unicode TR#16