NAME
>   Text::Iconv – Perl interface to iconv() codeset conversion function

SYNOPSIS
```
use Text::Iconv;
$converter = Text::Iconv->new("fromcode", "tocode");
$converted = $converter->convert("Text to convert");
```

DESCRIPTION
>   The **Text::Iconv** module provides a Perl interface to the **iconv()** function as defined by the Single UNIX Specification.

>   The **convert()** method converts the encoding of characters in the input string from the *fromcode* codeset to the *tocode* codeset, and returns the result.

>   Settings of *fromcode* and *tocode* and their permitted combinations are implementation-dependent. Valid values are specified in the system documentation; the **iconv** (1) utility should also provide a **−l** option that lists all supported codesets.

   Utility methods
>   **Text::Iconv** objects also provide the following methods:

>   **retval()** returns the return value of the underlying **iconv()** function for the last conversion; according to the Single UNIX Specification, this value indicates "the number of non-identical conversions performed." Note, however, that iconv implementations vary widely in the interpretation of this specification.

>   This method can be called after calling **convert()**, e.g.:

```
$result = $converter->convert("lorem ipsum dolor sit amet");
$retval = $converter->retval;
```

>   When called before the first call to **convert()**, or if an error occured during the conversion, **retval()** returns **undef**.

>   **get_attr()**: This method is only available with GNU libiconv, otherwise it throws an exception. The **get_attr()** method allows you to query various attributes which influence the behavior of **convert()**. The currently supported attributes are *trivialp*, *transliterate*, and *discard_ilseq*, e.g.:

```
$state = $converter->get_attr("transliterate");
```

>   See **iconvctl** (3) for details. To ensure portability to other iconv implementations you should first check for the availability of this method using **eval {}**, e.g.:

```
eval { $conv->get_attr("trivialp") };
if ($@)
{
  # get_attr() is not available
}
else
{
  # get_attr() is available
}
```

>   This method should be considered experimental.

>   **set_attr()**: This method is only available with GNU libiconv, otherwise it throws an exception. The **set_attr()** method allows you to set various attributes which influence the behavior of **convert()**. The currently supported attributes are *transliterate* and *discard_ilseq*, e.g.:

```
$state = $converter->set_attr("transliterate");
```

>   See **iconvctl** (3) for details. To ensure portability to other iconv implementations you should first check for the availability of this method using **eval {}**, cf. the description of **set_attr()** above.

>   This method should be considered experimental.

ERRORS

If the conversion can't be initialized an exception is raised (using **croak()**).

Handling of conversion errors

Text::Iconv provides a class attribute **raise_error** and a corresponding class method for setting and getting its value. The handling of errors during conversion depends on the setting of this attribute. If **raise_error** is set to a true value, an exception is raised; otherwise, the **convert()** method only returns **undef**. By default **raise_error** is false. Example usage:

```
Text::Iconv->raise_error(1);    # Conversion errors raise exceptions
Text::Iconv->raise_error(0);    # Conversion errors return undef
$a = Text::Iconv->raise_error(); # Get current setting
```

Per-object handling of conversion errors

As an experimental feature, *Text::Iconv* also provides an instance attribute **raise_error** and a corresponding method for setting and getting its value. If **raise_error** is **undef**, the class-wide settings apply. If **raise_error** is 1 or 0 (true or false), the object settings override the class-wide settings.

Consult **iconv** (3) for details on errors that might occur.

Conversion of undef

Converting **undef**, e.g.,

```
$converted = $converter->convert(undef);
```

always returns **undef**. This is not considered an error.

NOTES

The supported codesets, their names, the supported conversions, and the quality of the conversions are all system-dependent.

AUTHOR

Michael Piotrowski <mxp@dynalabs.de>

SEE ALSO

iconv (1), **iconv** (3)