NAME

Locale::Script - ISO codes for script identification (ISO 15924)

SYNOPSIS

```
use Locale::Script;
use Locale::Constants;

$script  = code2script('ph');                       # 'Phoenician'
$code    = script2code('Tibetan');                  # 'bo'
$code3   = script2code('Tibetan',
                       LOCALE_CODE_ALPHA_3);        # 'bod'
$codeN   = script2code('Tibetan',
                       LOCALE_CODE_ALPHA_NUMERIC);  # 330

@codes   = all_script_codes();
@scripts = all_script_names();
```

DESCRIPTION

The `Locale::Script` module provides access to the ISO codes for identifying scripts, as defined in ISO 15924. For example, Egyptian hieroglyphs are denoted by the two-letter code 'eg', the three-letter code 'egy', and the numeric code 050.

You can either access the codes via the conversion routines (described below), or with the two functions which return lists of all script codes or all script names.

There are three different code sets you can use for identifying scripts:

alpha-2

Two letter codes, such as 'bo' for Tibetan. This code set is identified with the symbol `LOCALE_CODE_ALPHA_2`.

alpha-3

Three letter codes, such as 'ell' for Greek. This code set is identified with the symbol `LOCALE_CODE_ALPHA_3`.

numeric

Numeric codes, such as 410 for Hiragana. This code set is identified with the symbol `LOCALE_CODE_NUMERIC`.

All of the routines take an optional additional argument which specifies the code set to use. If not specified, it defaults to the two-letter codes. This is partly for backwards compatibility (previous versions of Locale modules only supported the alpha-2 codes), and partly because they are the most widely used codes.

The alpha-2 and alpha-3 codes are not case-dependent, so you can use 'BO', 'Bo', 'bO' or 'bo' for Tibetan. When a code is returned by one of the functions in this module, it will always be lower-case.

SPECIAL CODES

The standard defines various special codes.

- The standard reserves codes in the ranges **qa** - **qt**, **qaa** - **qat**, and **900** - **919**, for private use.
- **zx**, **zxx**, and **997**, are the codes for unwritten languages.
- **zy**, **zyy**, and **998**, are the codes for an undetermined script.
- **zz**, **zzz**, and **999**, are the codes for an uncoded script.

The private codes are not recognised by Locale::Script, but the others are.

CONVERSION ROUTINES

There are three conversion routines: `code2script()`, `script2code()`, and `script_code2code()`.

code2script(CODE, [CODESET])

This function takes a script code and returns a string which contains the name of the script identified. If the code is not a valid script code, as defined by ISO 15924, then `undef` will be returned:

```
$script = code2script('cy');   # Cyrillic
```

script2code(STRING, [CODESET])

This function takes a script name and returns the corresponding script code, if such exists. If the argument could not be identified as a script name, then `undef` will be returned:

```
$code = script2code('Gothic', LOCALE_CODE_ALPHA_3);
# $code will now be 'gth'
```

The case of the script name is not important. See the section *KNOWN BUGS AND LIMITATIONS* below.

script_code2code(CODE, CODESET, CODESET)

This function takes a script code from one code set, and returns the corresponding code from another code set.

```
 $alpha2 = script_code2code('jwi',
LOCALE_CODE_ALPHA_3 => LOCALE_CODE_ALPHA_2);
 # $alpha2 will now be 'jw' (Javanese)
```

If the code passed is not a valid script code in the first code set, or if there isn't a code for the corresponding script in the second code set, then `undef` will be returned.

QUERY ROUTINES

There are two function which can be used to obtain a list of all codes, or all script names:

`all_script_codes ( [ CODESET ] )`

Returns a list of all two-letter script codes. The codes are guaranteed to be all lower-case, and not in any particular order.

`all_script_names ( [ CODESET ] )`

Returns a list of all script names for which there is a corresponding script code in the specified code set. The names are capitalised, and not returned in any particular order.

EXAMPLES

The following example illustrates use of the `code2script()` function. The user is prompted for a script code, and then told the corresponding script name:

```
$| = 1;   # turn off buffering

print "Enter script code: ";
chop($code = <STDIN>);
$script = code2script($code, LOCALE_CODE_ALPHA_2);
if (defined $script)
{
    print "$code = $script\n";
```

```
}
else
{
    print "'$code' is not a valid script code!\n";
}
```

KNOWN BUGS AND LIMITATIONS

- When using `script2code()`, the script name must currently appear exactly as it does in the source of the module. For example,

  ```
  script2code('Egyptian hieroglyphs')
  ```

  will return **eg**, as expected. But the following will all return `undef`:

  ```
  script2code('hieroglyphs')
  script2code('Egyptian Hieroglypics')
  ```

  If there's need for it, a future version could have variants for script names.

- In the current implementation, all data is read in when the module is loaded, and then held in memory. A lazy implementation would be more memory friendly.

SEE ALSO

Locale::Language

ISO two letter codes for identification of language (ISO 639).

Locale::Currency

ISO three letter codes for identification of currencies and funds (ISO 4217).

Locale::Country

ISO three letter codes for identification of countries (ISO 3166)

ISO 15924

The ISO standard which defines these codes.

http://www.evertype.com/standards/iso15924/

Home page for ISO 15924.

AUTHOR

Neil Bowers <neil@bowers.com>

COPYRIGHT

Copyright (c) 2002-2004 Neil Bowers.

This module is free software; you can redistribute it and/or modify it under the same terms as Perl itself.