# babelfish Documentation

***Release 0.5.5-dev***

**Antoine Bertin**

April 11, 2016

Contents


**1 Script** **3**

**2 Country** **5**

**3 Language** **7**

**4 Custom Converters** **9**

**5 API Documentation** **11**
5.1 Script . . . 11
5.2 Country . . . 11
5.3 Language . . . 12
5.4 Converter Bases . . . 13
5.5 Exceptions . . . 15

**6 Changelog** **17**
6.1 0.5.5 . . . 17
6.2 0.5.4 . . . 17
6.3 0.5.3 . . . 17
6.4 0.5.2 . . . 17
6.5 0.5.1 . . . 17
6.6 0.5.0 . . . 18
6.7 0.4.0 . . . 18
6.8 0.3.0 . . . 18
6.9 0.2.1 . . . 18
6.10 0.2.0 . . . 18
6.11 0.1.5 . . . 19
6.12 0.1.4 . . . 19
6.13 0.1.3 . . . 19
6.14 0.1.2 . . . 19
6.15 0.1.1 . . . 19
6.16 0.1 . . . 19

**Python Module Index** **21**

Release v0.5.5-dev

BabelFish is a Python library to work with countries and languages.

CHAPTER 1

# Script

Simple script representation from 4-letter code (ISO-15924):

```
>>> script = babelfish.Script('Hira')
>>> script
<Script [Hira]>
```

CHAPTER 2

# Country

Simple country representation from 2-letter code (ISO-3166):

```
>>> country = babelfish.Country('GB')
>>> country
<Country [GB]>
```

Built-in country converters (name):

```
>>> country.name
'UNITED KINGDOM'
```

CHAPTER 3

# Language

Simple language representation from 3-letter code (ISO-639-3):

```
>>> language = babelfish.Language('eng')
>>> language
<Language [en]>
```

Country specific language:

```
>>> language = babelfish.Language('por', 'BR')
>>> language
<Language [pt-BR]>
```

Language with specific script:

```
>>> language = babelfish.Language.fromalpha2('sr')
>>> language.script = babelfish.Script('Cyrl')
>>> language
<Language [sr-Cyrl]>
```

Built-in language converters (alpha2, alpha3b, alpha3t, name, scope, type and opensubtitles):

```
>>> language = babelfish.Language('por', 'BR')
>>> language.alpha2
'pt'
>>> language.scope
'individual'
>>> language.type
'living'
>>> language.opensubtitles
'pob'
>>> babelfish.Language.fromalpha3b('fre')
<Language [fr]>
```

CHAPTER 4

# Custom Converters

Build your own Language/Country converter:

```
class MyCodeConverter(babelfish.LanguageReverseConverter):
    def __init__(self):
        self.to_mycode = {'fra': 'mycode1', 'eng': 'mycode2'}
        self.from_mycode = {'mycode1': 'fra', 'mycode2': 'eng'}
    def convert(self, alpha3, country=None, script=None):
        if alpha3 not in self.to_mycode:
            raise babelfish.LanguageConvertError(alpha3, country, script)
        return self.to_mycode[alpha3]
    def reverse(self, mycode):
        if mycode not in self.from_mycode:
            raise babelfish.LanguageReverseError(mycode)
        return (self.from_mycode[mycode],)
```

You can also use the *LanguageEquivalenceConverter* utility class if your mapping is a simple one-to-one mapping:

```
class MyCodeConverter(babelfish.LanguageEquivalenceConverter):
    SYMBOLS = {'fra': 'mycode1', 'eng': 'mycode2'}
```

Use it directly (no lazy loading):

```
>>> babelfish.LANGUAGE_CONVERTERS['mycode'] = MyCodeConverter()
>>> babelfish.Language.frommycode('mycode2')
<Language [en]>
>>> babelfish.Language('fra').mycode
'mycode1'
```

Or make it available in your application by using the entry point (lazy loading):

```
setup([...],
      entry_points={'babelfish.language_converters': ['mycode = mymodule.converter:MyCodeConverter']
      [...])
```

Or if you don't want to use the entry point (lazy loading):

```
>>> babelfish.language_converters.register('mycode = mymodule.converter:MyCodeConverter')
```

CHAPTER 5

# API Documentation

If you are looking for information on a specific function, class or method, this part of the documentation is for you.

## 5.1 Script

babelfish.script.**SCRIPTS**
Dictionary of script ISO-15924 codes to English names

**class** babelfish.script.**Script**(*script*)
A human writing system

A script is represented by a 4-letter code from the ISO-15924 standard

**Parameters** **script** (*string*) – 4-letter ISO-15924 script code

**code = None**
ISO-15924 4-letter script code

**name**
English name of the script

## 5.2 Country

babelfish.country.**COUNTRIES**
Country code to country name mapping

babelfish.country.**COUNTRY_MATRIX**
List of countries in the ISO-3166-1 as namedtuple of alpha2 and name

**class** babelfish.country.**CountryConverterManager**
*ConverterManager* for country converters

babelfish.country.**COUNTRY_CONVERTERS**
Instance of *CountryConverterManager*

**class** babelfish.country.**CountryMeta**
The *Country* metaclass

Dynamically redirect `Country.frommycode()` to *Country.fromcode()* with the `mycode` *converter*

**class** babelfish.country.**Country**(*country*)
A country on Earth

A country is represented by a 2-letter code from the ISO-3166 standard

**Parameters** **country** (*string*) – 2-letter ISO-3166 country code

**alpha2 = None**
ISO-3166 2-letter country code

**classmethod fromcode**(*code*, *converter*)
Create a `Country` by its *code* using *converter* to `reverse()` it

**Parameters**

- **code** (*string*) – the code to reverse
- **converter** (*string*) – name of the `CountryReverseConverter` to use

**Returns** the corresponding `Country` instance

**Return type** `Country`

## 5.3 Language

babelfish.language.**LANGUAGES**
Available language codes

babelfish.language.**LANGUAGE_MATRIX**
List of languages in the ISO-639-3 as namedtuple of alpha3, alpha3b, alpha3t, alpha2, scope, type, name and comment

**class** babelfish.language.**LanguageConverterManager**
`ConverterManager` for language converters

babelfish.language.**LANGUAGE_CONVERTERS**
Instance of `LanguageConverterManager`

**class** babelfish.language.**LanguageMeta**
The `Language` metaclass

Dynamically redirect `Language.frommycode()` to `Language.fromcode()` with the `mycode` *converter*

**class** babelfish.language.**Language**(*language*, *country=None*, *script=None*, *unknown=None*)
A human language

A human language is composed of a language part following the ISO-639 standard and can be country-specific when a `Country` is specified.

The `Language` is extensible with custom converters (see *Custom Converters*)

**Parameters**

- **language** (*string*) – the language as a 3-letter ISO-639-3 code
- **country** (string or `Country` or None) – the country (if any) as a 2-letter ISO-3166 code or `Country` instance
- **script** (string or `Script` or None) – the script (if any) as a 4-letter ISO-15924 code or `Script` instance
- **unknown** (*string or None*) – the unknown language as a three-letters ISO-639-3 code to use as fallback

**Raise** ValueError if the language could not be recognized and *unknown* is `None`

**classmethod fromcode**(*code*, *converter*)
Create a `Language` by its *code* using *converter* to `reverse()` it

**Parameters**

- **code** (*string*) – the code to reverse
- **converter** (*string*) – name of the `LanguageReverseConverter` to use

**Returns** the corresponding `Language` instance

**Return type** `Language`

**classmethod fromietf**(*ietf*)
Create a `Language` by from an IETF language code

**Parameters** **ietf** (*string*) – the ietf code

**Returns** the corresponding `Language` instance

**Return type** `Language`

## 5.4 Converter Bases

**class** babelfish.converters.**LanguageConverter**
A `LanguageConverter` supports converting an alpha3 language code with an alpha2 country code and a script code into a custom code

**codes**
Set of possible custom codes

**convert**(*alpha3*, *country=None*, *script=None*)
Convert an alpha3 language code with an alpha2 country code and a script code into a custom code

**Parameters**

- **alpha3** (*string*) – ISO-639-3 language code
- **country** (*string or None*) – ISO-3166 country code, if any
- **script** (*string or None*) – ISO-15924 script code, if any

**Returns** the corresponding custom code

**Return type** string

**Raise** `LanguageConvertError`

**class** babelfish.converters.**LanguageReverseConverter**
A `LanguageConverter` able to reverse a custom code into a alpha3 ISO-639-3 language code, alpha2 ISO-3166-1 country code and ISO-15924 script code

**reverse**(*code*)
Reverse a custom code into alpha3, country and script code

**Parameters** **code** (*string*) – custom code to reverse

**Returns** the corresponding alpha3 ISO-639-3 language code, alpha2 ISO-3166-1 country code and ISO-15924 script code

**Return type** tuple

**Raise** `LanguageReverseError`

**class** babelfish.converters.**LanguageEquivalenceConverter**

A *LanguageEquivalenceConverter* is a utility class that allows you to easily define a *LanguageReverseConverter* by only specifying the dict from alpha3 to their corresponding symbols.

You must specify the dict of equivalence as a class variable named SYMBOLS.

If you also set the class variable CASE_SENSITIVE to `True` then the reverse conversion function will be case-sensitive (it is case-insensitive by default).

Example:

```
class MyCodeConverter(babelfish.LanguageEquivalenceConverter):
    CASE_SENSITIVE = True
    SYMBOLS = {'fra': 'mycode1', 'eng': 'mycode2'}
```

**class** babelfish.converters.**CountryConverter**

A *CountryConverter* supports converting an alpha2 country code into a custom code

**codes**

Set of possible custom codes

**convert**(*alpha2*)

Convert an alpha2 country code into a custom code

**Parameters** **alpha2** (*string*) – ISO-3166-1 language code

**Returns** the corresponding custom code

**Return type** string

**Raise** *CountryConvertError*

**class** babelfish.converters.**CountryReverseConverter**

A *CountryConverter* able to reverse a custom code into a alpha2 ISO-3166-1 country code

**reverse**(*code*)

Reverse a custom code into alpha2 code

**Parameters** **code** (*string*) – custom code to reverse

**Returns** the corresponding alpha2 ISO-3166-1 country code

**Return type** string

**Raise** *CountryReverseError*

**class** babelfish.converters.**ConverterManager**

Manager for babelfish converters behaving like a dict with lazy loading

Loading is done in this order:

- Entry point converters
- Registered converters
- Internal converters

**entry_point**

The entry point where to look for converters

**internal_converters**

Internal converters with entry point syntax

**registered_converters** = **None**

Registered converters with entry point syntax

**converters** = None
Loaded converters

**register**(*entry_point*)
Register a converter

**Parameters** **entry_point** (*string*) – converter to register (entry point syntax)

**Raise** ValueError if already registered

**unregister**(*entry_point*)
Unregister a converter

**Parameters** **entry_point** (*string*) – converter to unregister (entry point syntax)

## 5.5 Exceptions

**class** babelfish.exceptions.**Error**
Base class for all exceptions in babelfish

**class** babelfish.exceptions.**LanguageConvertError**(*alpha3*, *country=None*, *script=None*)
Exception raised by converters when `convert()` fails

**Parameters**

- **alpha3** (*string*) – alpha3 code that failed conversion
- **country** (*string or None*) – country code that failed conversion, if any
- **script** (*string or None*) – script code that failed conversion, if any

**class** babelfish.exceptions.**LanguageReverseError**(*code*)
Exception raised by converters when `reverse()` fails

**Parameters** **code** (*string*) – code that failed reverse conversion

**class** babelfish.exceptions.**CountryConvertError**(*alpha2*)
Exception raised by converters when `convert()` fails

**Parameters** **alpha2** (*string*) – alpha2 code that failed conversion

**class** babelfish.exceptions.**CountryReverseError**(*code*)
Exception raised by converters when `reverse()` fails

**Parameters** **code** (*string*) – code that failed reverse conversion

CHAPTER 6

# Changelog

## 6.1 0.5.5

**release date:** 2015-10-31

- Fix hasattr on Country object when called with an invalid attribute

## 6.2 0.5.4

**release date:** 2015-01-24

- Fix setuptools deprecation warning

## 6.3 0.5.3

**release date:** 2014-06-22

- Better equality semantics for Language, Country, Script

## 6.4 0.5.2

**release date:** 2014-05-25

- Babelfish objects (Language, Country, Script) are now picklable
- Added support for Python 3.4

## 6.5 0.5.1

**release date:** 2014-01-26

- Add a register method to ConverterManager to register without loading

## 6.6 0.5.0

**release date:** 2014-01-25

**WARNING:** Backward incompatible changes

- Simplify converter management with ConverterManager class
- Make babelfish usable in place
- Add Python 2.6 / 3.2 compatibility

## 6.7 0.4.0

**release date:** 2013-11-21

**WARNING:** Backward incompatible changes

- Add converter support for Country
- Language/country reverse name detection is now case-insensitive
- Add alpha3t, scope and type converters
- Use lazy loading of converters

## 6.8 0.3.0

**release date:** 2013-11-09

- Add support for scripts
- Improve built-in converters
- Add support for ietf

## 6.9 0.2.1

**release date:** 2013-11-03

- Fix reading of data files

## 6.10 0.2.0

**release date:** 2013-10-31

- Add str method
- More explicit exceptions
- Change repr format to use ascii only

## 6.11 0.1.5

**release date:** 2013-10-21

- Add a fromcode method on Language class
- Add a codes attribute on converters

## 6.12 0.1.4

**release date:** 2013-10-20

- Fix converters not raising NoConversionError

## 6.13 0.1.3

**release date:** 2013-09-29

- Fix source distribution

## 6.14 0.1.2

**release date:** 2013-09-29

- Add missing files to source distribution

## 6.15 0.1.1

**release date:** 2013-09-28

- Fix python3 support

## 6.16 0.1

**release date:** 2013-09-28

- Initial version

## b

babelfish.converters, 13
babelfish.country, 11
babelfish.exceptions, 15
babelfish.language, 12
babelfish.script, 11

## A

alpha2 (babelfish.country.Country attribute), 12

## B

babelfish.converters (module), 13
babelfish.country (module), 11
babelfish.exceptions (module), 15
babelfish.language (module), 12
babelfish.script (module), 11

## C

code (babelfish.script.Script attribute), 11
codes (babelfish.converters.CountryConverter attribute), 14
codes (babelfish.converters.LanguageConverter attribute), 13
convert() (babelfish.converters.CountryConverter method), 14
convert() (babelfish.converters.LanguageConverter method), 13
ConverterManager (class in babelfish.converters), 14
converters (babelfish.converters.ConverterManager attribute), 14
COUNTRIES (in module babelfish.country), 11
Country (class in babelfish.country), 11
COUNTRY_CONVERTERS (in module babelfish.country), 11
COUNTRY_MATRIX (in module babelfish.country), 11
CountryConverter (class in babelfish.converters), 14
CountryConverterManager (class in babelfish.country), 11
CountryConvertError (class in babelfish.exceptions), 15
CountryMeta (class in babelfish.country), 11
CountryReverseConverter (class in babelfish.converters), 14
CountryReverseError (class in babelfish.exceptions), 15

## E

entry_point (babelfish.converters.ConverterManager attribute), 14
Error (class in babelfish.exceptions), 15

## F

fromcode() (babelfish.country.Country class method), 12
fromcode() (babelfish.language.Language class method), 12
fromietf() (babelfish.language.Language class method), 13

## I

internal_converters (babelfish.converters.ConverterManager attribute), 14

## L

Language (class in babelfish.language), 12
LANGUAGE_CONVERTERS (in module babelfish.language), 12
LANGUAGE_MATRIX (in module babelfish.language), 12
LanguageConverter (class in babelfish.converters), 13
LanguageConverterManager (class in babelfish.language), 12
LanguageConvertError (class in babelfish.exceptions), 15
LanguageEquivalenceConverter (class in babelfish.converters), 13
LanguageMeta (class in babelfish.language), 12
LanguageReverseConverter (class in babelfish.converters), 13
LanguageReverseError (class in babelfish.exceptions), 15
LANGUAGES (in module babelfish.language), 12

## N

name (babelfish.script.Script attribute), 11

## R

register() (babelfish.converters.ConverterManager method), 15
registered_converters (babelfish.converters.ConverterManager attribute), 14

reverse() (babelfish.converters.CountryReverseConverter method), 14
reverse() (babelfish.converters.LanguageReverseConverter method), 13

## S

Script (class in babelfish.script), 11
SCRIPTS (in module babelfish.script), 11

## U

unregister() (babelfish.converters.ConverterManager method), 15