
Aruudy Documentation

Release 0.5.1

Abdelkrime Aries

Dec 29, 2022

Contents

1	Functions	3
1.1	normalize	3
1.2	process_shatr	3
1.3	prosody_form	4
2	Classes	5
2.1	Bayt	5
2.2	Shatr	5
3	aruudy.poetry.meter Module	7
3.1	Functions	7
3.2	Classes	9
4	aruudy.poetry.foot Module	15
4.1	Classes	15
	Python Module Index	21
	Index	23

This module is used for prosody tasks:

- normalize a text
- find prosody form of a text
- find the meter

<code>normalize(text)</code>	Normalize a given text.
<code>process_shatr(text)</code>	process a shatr.
<code>prosody_form(text)</code>	Generate the prosody form.

1.1 normalize

`aruudy.poetry.prosody.normalize(text)`

Normalize a given text.

The text is normalized by:

- deleting non Arabic characters.
- Deleting Tatweel (a character used as ligature)
- Vocalizing and correcting letters

Parameters `text` (*str*) – The text to be normalized.

Returns A normalized text.

Return type *str*

1.2 process_shatr

`aruudy.poetry.prosody.process_shatr(text)`

process a shatr.

Parameters `text` (*str*) – The text representing the shatr (part of verse).

Returns The Shatr containing information about the text.

Return type *Shatr*

1.3 prosody_form

`aruudy.poetry.prosody.prosody_form(text)`

Generate the prosody form.

The prosody form is generated by:

- deleting some letters such as al-
- adding some letters; for example, shadda.

Parameters `text` (*str*) – A normal vocalized Arabic text.

Returns The text in its prosody form.

Return type `str`

`Bayt(text[, sep])`

`Shatr(text)`

 Shatr (Hemistich) object.

2.1 Bayt

```
class aruudy.poetry.prosody.Bayt (text, sep='r')
    Bases: object
```

2.2 Shatr

```
class aruudy.poetry.prosody.Shatr (text)
    Bases: object
```

Shatr (Hemistich) object.

In Arabic, a verse is composed of two hemistiches (shatr). This object will contain all information about a given text.

Parameters `text` (`str`) – The text to be analysed.

orig
the original input text.

Type `str`

norm
the text after normalization.

Type `str`

prosody
prosody form of a text.

Type `str`

ameter

the Arabic meter using “w” as watad (peg) and “s” as sabab (cord).

Type `str`

emeter

the English meter using “-” for long syllables and “u” for short ones.

Type `str`

bahr

The bahr (meter) of the input text.

Type `Bahr`

parts

The text split into different parts. Each part is a dict with these components:

- **emeter**: the english meter of that part
- **type**: the type of that part: is it regular or else
- **part**: the text of that part
- **mnemonic**: the mnemonic text of that part

Type `list(Part)`

Methods Summary

<code>to_dict([bahr, parts])</code>	Generates a dict object of the Bahr.
-------------------------------------	--------------------------------------

Methods Documentation

to_dict (*bahr=False, parts=False*)

Generates a dict object of the Bahr.

Parameters

- **bahr** (*bool*) – if true: the bahr object will be dictionarized too. By default, it is False. Hence, the “bahr” is an object of type Bahr
- **parts** (*bool*) – if true: the list of parts will be dictionarized too. By default, it is False. Hence, the parts in the list are objects of type Part

Returns

A dictionary object describing the Shatr.

- **norm** (*str*): normalized text
- **prosody** (*str*): prosody form of the text
- **ameter** (*str*): Arabic meter of the text
- **emeter** (*str*): English meter of the text
- **bahr** (*Bahr or dict*): the bahr of the text
- **parts** (*list(Part or dict)*): the parts of the text

Return type `dict`

3.1 Functions

<code>a2e_meter(ameter)</code>	Transforms an Arabic meter to an English one.
<code>e2a_meter(emeter)</code>	Transforms an English meter to an Arabic one.
<code>extract_meter(bahrForm[, used])</code>	Extract the meter description from a list of <i>Tafiila</i> objects.
<code>get_ameter(text)</code>	Get the Arabic meter of a given text.
<code>get_bahr(name[, dic])</code>	Search for poetry Bahr by name.
<code>get_names([lang])</code>	get a list of meters names.
<code>name_type(name)</code>	decides if a name is in English or Arabic.
<code>search_bahr(emeter[, units])</code>	Search for Bahr of a given English meter.

3.1.1 a2e_meter

`aruudy.poetry.meter.a2e_meter(ameter)`

Transforms an Arabic meter to an English one. The Arabic meter uses vocalization as a basis:

- “w” watad (peg) which are vocalized letters
- “s” sabab (cord) which are vowels and unvocalized letters

While English meter uses syllables:

- “-” for long syllables, equivalent to “ws” in the Arabic one
- “u” for short syllables, equivalent to “w” in the Arabic one.

Parameters `ameter` (*str*) – The Arabic meter using the two letters: “w” and “s”.

Returns The English meter using the two characters: “-” and “u”.

Return type *str*

3.1.2 e2a_meter

`aruudy.poetry.meter.e2a_meter` (*emeter*)

Transforms an English meter to an Arabic one. The English meter uses syllables as a basis:

- “-” for long syllables, equivalent to “ws” in the Arabic one
- “u” for short syllables, equivalent to “w” in the Arabic one.

While the Arabic meter uses vocalization:

- “w” watad (peg) which are vocalized letters
- “s” sabab (cord) which are vowels and unvocalized letters

Parameters `emeter` (*str*) – The English meter using the two characters: “-” and “u”.

Returns The Arabic meter using the two letters: “w” and “s”.

Return type `str`

3.1.3 extract_meter

`aruudy.poetry.meter.extract_meter` (*bahrForm*, *used=True*)

Extract the meter description from a list of *Tafiila* objects.

Parameters

- **bahrForm** (*BahrForm*) – An object describing the meter’s form.
- **used** (*bool*) – Meters, in Arabic, can have used forms different than standard ones. if True: the result is used form. Otherwise, it is standard form

Returns

A dictionary object describing the meter represented by the feet. The dictionary contains these elements:

- **type**: a string describing the type of each foot (tafila)
- **mnemonic**: a string describing the mnemonic of each foot.
- **emeter**: a string describing the English meter of each foot.
- **ameter**: a string describing the Arabic meter of each foot.

Return type `dict`

3.1.4 get_ameter

`aruudy.poetry.meter.get_ameter` (*text*)

Get the Arabic meter of a given text. Produces the Arabic meter of a given text in prosody form. The Arabic meter is composed of two letters:

- “w” watad (peg) which are vocalized letters
- “s” sabab (cord) which are vowels and unvocalized letters

Parameters `text` (*str*) – Arabic text in prosody form.

Returns Arabic meter of the input text. A string composed of “w” and “s”.

Return type `str`

3.1.5 get_bahr

`aruudy.poetry.meter.get_bahr(name, dic=True)`

Search for poetry Bahr by name.

Parameters

- **name** (*str*) – name of the poetry Bahr (meter).
- **dic** (*bool*) – True(default): it returns a dict object with all information. If False, it returns an object of type Bahr

Returns dict: containing the information. or a Bahr object. or None

Return type *dict*

3.1.6 get_names

`aruudy.poetry.meter.get_names(lang=None)`

get a list of meters names.

Parameters **lang** (*str*) – If not specified: the result will be all available names.

Returns A list of names.

Return type *list(Union[str, dict])*

3.1.7 name_type

`aruudy.poetry.meter.name_type(name)`

decides if a name is in English or Arabic.

Parameters **name** (*str*) – The name we want to test.

Returns “english” or “arabic”.

Return type *str*

3.1.8 search_bahr

`aruudy.poetry.meter.search_bahr(emeter, units=[])`

Search for Bahr of a given English meter.

Parameters

- **emeter** (*str*) – English meter.
- **units** (*list(str)*) – A list of vocalized and unvocalized elements, generated from the text.

Returns A tuple of the found meter and the part’s description. If not found, it will return (None, None)

Return type *tuple(Bahr, Part)*

3.2 Classes

<code>Bahr</code> (info)	Representation of the Arabic meter.
<code>BahrForm</code> (feet)	The form of a Bahr (meter).
FT	alias of <code>aruudy.poetry.foot.TafiilaType</code>
<code>Part</code> (tafiila_comp)	The text's part description.
<code>TafiilaComp</code> (comp)	Tafiila (foot) components.

3.2.1 Bahr

class `aruudy.poetry.meter.Bahr` (info)

Bases: `object`

Representation of the Arabic meter.

Parameters `info` (*dict*) – Description of parameter *info*.

name

Bahr's name, which is composed of:

- arabic: its name in Arabic
- english: its name in English
- trans: its Arabic name's transliteration.

Type `dict`

used_scansion

The most used scansion. The dictionary contains these elements:

- type: a string describing the type of each foot (tafiila)
- mnemonic: a string describing the mnemonic of each foot.
- emeter: a string describing the English meter of each foot.
- ameter: a string describing the Arabic meter of each foot.

Type `dict`

meter

A list of meter's forms.

Type `list(BahrForm)`

std_scansion

the standard scansion. The dictionary contains these elements:

- type: a string describing the type of each foot (tafiila)
- mnemonic: a string describing the mnemonic of each foot.
- emeter: a string describing the English meter of each foot.
- ameter: a string describing the Arabic meter of each foot.

Type `dict`

Methods Summary

<code>get_names()</code>	Get the names of the meter.
<code>test_name(key, value)</code>	Test if .
<code>to_dict()</code>	Transform the bahr to a dictionary.
<code>validate(emeter[, units])</code>	Validate a given emeter into one of the forms.

Methods Documentation

`get_names()`

Get the names of the meter.

Returns

Bahr's name, which is composed of:

- **arabic**: its name in Arabic
- **english**: its name in English
- **trans**: its Arabic name's transliteration.

Return type `dict`

`test_name(key, value)`

Test if .

Parameters

- **key** (`str`) – can be “arabic”, “english” or “trans”.
- **value** (`str`) – The name we are looking for.

Returns True, if this meter have the name specified by “value”

Return type `bool`

`to_dict()`

Transform the bahr to a dictionary.

Returns The dictionary has three components “name”, “used_scansion” and “std_scansion” which are dictionaries too. They are described in the attributes section.

Return type `dict`

`validate(emeter, units=[])`

Validate a given emeter into one of the forms.

Search for a form which the given emeter follows.

Parameters

- **emeter** (`str`) – English meter.
- **units** (`list(str)`) – A list of vocalized and unvocalized elements, generated from the text.

Returns the part object.

Return type `Part`

3.2.2 BahrForm

class arudy.poetry.meter.**BahrForm**(*feet*)

Bases: `object`

The form of a Bahr (meter).

For a given Arabic meter (Bahr), there may be multiple forms.

Parameters **feet** (`list(Tafiila)`) – A list of feet describing the meter.

feet

A list of feet describing the meter.

Type `list(Tafiila)`

Methods Summary

<code>validate(emeter[, units])</code>	Checks if an emeter follows this meter's form.
--	--

Methods Documentation

validate (*emeter*, *units=[]*)

Checks if an emeter follows this meter's form.

Parameters

- **emeter** (`str`) – The English meter of the text.
- **units** (`list(str)`) – A list of vocalized and unvocalized elements, generated from the text.

Returns The part object.

Return type `Part`

3.2.3 Part

class arudy.poetry.meter.**Part**(*tafiila_comp*)

Bases: `arudy.poetry.foot.TafiilaComp`

The text's part description.

Parameters **tafiila_comp** (`TafiilaComp`) – The description of the Foot which this part is based on.

ameter

The Arabic meter.

Type `str`

emeter

The english meter.

Type `type`

text

The part of text following that meter.

Type `type`

Methods Summary

<code>extract([units])</code>	Extracts the part of text following the meter.
<code>to_dict()</code>	Transforms this object to a dictionary.

Methods Documentation

extract (*units*=[])

Extracts the part of text following the meter.

Parameters **units** (*list* (*str*)) – A list of vocalized and unvocalized elements, generated from the text.

Returns the text following the meter.

Return type *str*

to_dict ()

Transforms this object to a dictionary.

Returns

The dictionary will contain:

- *type* (*TafilaComp*): the type of the foot
- *emeter* (*str*): the English meter
- *ameter* (*str*): the Arabic meter
- *mnemonic* (*str*): the mnemonic describing the meter
- *text* (*str*): the text following the meter

Return type *dict*

4.1 Classes

<i>Tafiila</i>	A class describing the foot.
<i>TafiilaComp</i> (comp)	Tafiila (foot) components.
<i>TafiilaType</i>	A class with different anomalies (Zuhaf and Illa) happening to the Foot.
<i>WSWSWSW</i> ([var])	
<i>WSWSWWS</i> ([var])	
<i>WSWWS</i> ([var])	
<i>WSWWSWS</i> ([var])	
<i>WWSWS</i> ([var])	
<i>WWSWSWS</i> ([var])	
<i>WWSWWWS</i> ([var])	
<i>WWWWSWS</i> ([var])	
<i>ZuhafIlla</i> (the_id, ar)	Short summary.

4.1.1 Tafiila

class aruudy.poetry.foot.**Tafiila**

Bases: `object`

A class describing the foot.

Methods Summary

<i>get_form</i> ([used])	Get the form of this Tafiila (foot).
<i>process</i> (text_emeter)	Process a given emeter to decide if it starts with this foot.

Methods Documentation

get_form (*used=True*)

Get the form of this Tafiila (foot).

Parameters **used** (*bool*) – if True, it will return the used form. Otherwise, it will return the standard one.

Returns The from describing this Tafiila (foot).

Return type *TafiilaComp*

process (*text_emeter*)

Process a given emeter to decide if it starts with this foot.

Parameters **text_emeter** (*str*) – The English meter of the text we want to process.

Returns

If the text's emeter starts with one of this foot's forms, this function will return a tuple of:

- The form (TafiilaComp) which the emeter starts with.
- The rest of the emeter

Otherwise, it returns (None, None)

Return type *tuple(TafiilaComp, str)*

4.1.2 TafiilaComp

class aruudy.poetry.foot.**TafiilaComp** (*comp*)

Bases: *object*

Tafiila (foot) components.

Parameters **comp** (*dict*) – A dictionary of the forme:

- type (TafiilaType): the type of the foot
- mnemonic (str): the mnemonic describing the foot
- emeter (str): the English meter

type

the type of the foot.

Type *TafiilaType*

mnemonic

the mnemonic describing the foot.

Type *str*

emeter

the English meter.

Type *str*

Methods Summary

<code>copy()</code>	Create a copy of this object.
---------------------	-------------------------------

Methods Documentation

copy()

Create a copy of this object.

Returns A copy of this object.

Return type *TafiilaComp*

4.1.3 TafiilaType

class aruudy.poetry.foot.**TafiilaType**

Bases: *object*

A class with different anomalies (Zuhaf and Illa) happening to the Foot.

This class can be considered as an enum.

Attributes Summary

AQL

ASAB

BATR

HADAD

HADF

IDHMAR

ISBAGH

KAFF

KASHF

KHABL

KHABN

KHAZL

NAQS

QABDH

QASR

QATE

QATF

SALAM

SALIM

SALIM type

SHAKL

TADIIL

TAI

TARFIIL

WAQS

Attributes Documentation

AQL = **AQL**

ASAB = **ASAB**

```

BATR = BATR
HADAD = HADAD
HADF = HADF
IDHMAR = IDHMAR
ISBAGH = ISBAGH
KAFF = KAFF
KASHF = KASHF
KHABL = KHABL
KHABN = KHABN
KHAZL = KHAZL
NAQS = NAQS
QABDH = QABDH
QASR = QASR
QATE = QATE
QATF = QATF
SALAM = SALAM
SALIM = SALIM
    SALIM type
SHAKL = SHAKL
TADIIL = TADIIL
TAI = TAI
TARFIIL = TARFIIL
WAQS = WAQS

```

4.1.4 WSWSWSW

```

class aruudy.poetry.foot.WSWSWSW (var=[SALIM])
    Bases: aruudy.poetry.foot.Tafiila

```

4.1.5 WSWSWWS

```

class aruudy.poetry.foot.WSWSWWS (var=[SALIM])
    Bases: aruudy.poetry.foot.Tafiila

```

4.1.6 WSWWS

```

class aruudy.poetry.foot.WSWWS (var=[SALIM])
    Bases: aruudy.poetry.foot.Tafiila

```

4.1.7 WSWWSWS

class aruudy.poetry.foot.**WSWWSWS** (*var=[SALIM]*)
 Bases: *aruudy.poetry.foot.Tafiila*

4.1.8 WWSWS

class aruudy.poetry.foot.**WWSWS** (*var=[SALIM]*)
 Bases: *aruudy.poetry.foot.Tafiila*

4.1.9 WWSWSWS

class aruudy.poetry.foot.**WWSWSWS** (*var=[SALIM]*)
 Bases: *aruudy.poetry.foot.Tafiila*

4.1.10 WWSWWWS

class aruudy.poetry.foot.**WWSWWWS** (*var=[SALIM]*)
 Bases: *aruudy.poetry.foot.Tafiila*

4.1.11 WWWSWWS

class aruudy.poetry.foot.**WWWSWWS** (*var=[SALIM]*)
 Bases: *aruudy.poetry.foot.Tafiila*

4.1.12 ZuhafIlla

class aruudy.poetry.foot.**ZuhafIlla** (*the_id, ar*)
 Bases: *object*

Short summary.

Parameters

- **the_id** (*str*) – The ID of the type (anomaly).
- **ar** (*str*) – The name of the type in Arabic.

id

The ID of the type (anomaly)..

Type *str*

ar

The name of the type in Arabic.

Type *str*

a

`aruudy.poetry.foot`, [15](#)
`aruudy.poetry.meter`, [7](#)
`aruudy.poetry.prosody`, [??](#)

A

`a2e_meter()` (in module *aruudy.poetry.meter*), 7
`ameter` (*aruudy.poetry.meter.Part* attribute), 12
`ameter` (*aruudy.poetry.prosody.Shatr* attribute), 6
`AQL` (*aruudy.poetry.foot.TafiilaType* attribute), 17
`ar` (*aruudy.poetry.foot.ZuhafIlla* attribute), 19
`aruudy.poetry.foot` (module), 15
`aruudy.poetry.meter` (module), 7
`aruudy.poetry.prosody` (module), 1
`ASAB` (*aruudy.poetry.foot.TafiilaType* attribute), 17

B

`bahr` (*aruudy.poetry.prosody.Shatr* attribute), 6
`Bahr` (class in *aruudy.poetry.meter*), 10
`BahrForm` (class in *aruudy.poetry.meter*), 12
`BATR` (*aruudy.poetry.foot.TafiilaType* attribute), 17
`Bayt` (class in *aruudy.poetry.prosody*), 5

C

`copy()` (*aruudy.poetry.foot.TafiilaComp* method), 17

E

`e2a_meter()` (in module *aruudy.poetry.meter*), 8
`emeter` (*aruudy.poetry.foot.TafiilaComp* attribute), 16
`emeter` (*aruudy.poetry.meter.Part* attribute), 12
`emeter` (*aruudy.poetry.prosody.Shatr* attribute), 6
`extract()` (*aruudy.poetry.meter.Part* method), 13
`extract_meter()` (in module *aruudy.poetry.meter*), 8

F

`feet` (*aruudy.poetry.meter.BahrForm* attribute), 12

G

`get_ameter()` (in module *aruudy.poetry.meter*), 8
`get_bahr()` (in module *aruudy.poetry.meter*), 9
`get_form()` (*aruudy.poetry.foot.Tafiila* method), 16
`get_names()` (*aruudy.poetry.meter.Bahr* method), 11
`get_names()` (in module *aruudy.poetry.meter*), 9

H

`HADAD` (*aruudy.poetry.foot.TafiilaType* attribute), 18
`HADF` (*aruudy.poetry.foot.TafiilaType* attribute), 18

I

`id` (*aruudy.poetry.foot.ZuhafIlla* attribute), 19
`IDHMAR` (*aruudy.poetry.foot.TafiilaType* attribute), 18
`ISBAGH` (*aruudy.poetry.foot.TafiilaType* attribute), 18

K

`KAFF` (*aruudy.poetry.foot.TafiilaType* attribute), 18
`KASHF` (*aruudy.poetry.foot.TafiilaType* attribute), 18
`KHABL` (*aruudy.poetry.foot.TafiilaType* attribute), 18
`KHABN` (*aruudy.poetry.foot.TafiilaType* attribute), 18
`KHAZL` (*aruudy.poetry.foot.TafiilaType* attribute), 18

M

`meter` (*aruudy.poetry.meter.Bahr* attribute), 10
`mnemonic` (*aruudy.poetry.foot.TafiilaComp* attribute), 16

N

`name` (*aruudy.poetry.meter.Bahr* attribute), 10
`name_type()` (in module *aruudy.poetry.meter*), 9
`NAQS` (*aruudy.poetry.foot.TafiilaType* attribute), 18
`norm` (*aruudy.poetry.prosody.Shatr* attribute), 5
`normalize()` (in module *aruudy.poetry.prosody*), 3

O

`orig` (*aruudy.poetry.prosody.Shatr* attribute), 5

P

`Part` (class in *aruudy.poetry.meter*), 12
`parts` (*aruudy.poetry.prosody.Shatr* attribute), 6
`process()` (*aruudy.poetry.foot.Tafiila* method), 16
`process_shatr()` (in module *aruudy.poetry.prosody*), 3
`prosody` (*aruudy.poetry.prosody.Shatr* attribute), 5

`prosody_form()` (in module `aruudy.poetry.prosody`),
4

Q

`QABDH` (*aruudy.poetry.foot.TafiilaType* attribute), 18

`QASR` (*aruudy.poetry.foot.TafiilaType* attribute), 18

`QATE` (*aruudy.poetry.foot.TafiilaType* attribute), 18

`QATF` (*aruudy.poetry.foot.TafiilaType* attribute), 18

S

`SALAM` (*aruudy.poetry.foot.TafiilaType* attribute), 18

`SALIM` (*aruudy.poetry.foot.TafiilaType* attribute), 18

`search_bahr()` (in module `aruudy.poetry.meter`), 9

`SHAKL` (*aruudy.poetry.foot.TafiilaType* attribute), 18

`Shatr` (class in `aruudy.poetry.prosody`), 5

`std_scansion` (*aruudy.poetry.meter.Bahr* attribute),
10

T

`TADIIL` (*aruudy.poetry.foot.TafiilaType* attribute), 18

`Tafiila` (class in `aruudy.poetry.foot`), 15

`TafiilaComp` (class in `aruudy.poetry.foot`), 16

`TafiilaType` (class in `aruudy.poetry.foot`), 17

`TAI` (*aruudy.poetry.foot.TafiilaType* attribute), 18

`TARFIIL` (*aruudy.poetry.foot.TafiilaType* attribute), 18

`test_name()` (*aruudy.poetry.meter.Bahr* method), 11

`text` (*aruudy.poetry.meter.Part* attribute), 12

`to_dict()` (*aruudy.poetry.meter.Bahr* method), 11

`to_dict()` (*aruudy.poetry.meter.Part* method), 13

`to_dict()` (*aruudy.poetry.prosody.Shatr* method), 6

`type` (*aruudy.poetry.foot.TafiilaComp* attribute), 16

U

`used_scansion` (*aruudy.poetry.meter.Bahr* attribute),
10

V

`validate()` (*aruudy.poetry.meter.Bahr* method), 11

`validate()` (*aruudy.poetry.meter.BahrForm* method),
12

W

`WAQS` (*aruudy.poetry.foot.TafiilaType* attribute), 18

`WSWSWSW` (class in `aruudy.poetry.foot`), 18

`WSWSWWS` (class in `aruudy.poetry.foot`), 18

`WSWWS` (class in `aruudy.poetry.foot`), 18

`WSWWSWS` (class in `aruudy.poetry.foot`), 19

`WWSWS` (class in `aruudy.poetry.foot`), 19

`WWSWSWS` (class in `aruudy.poetry.foot`), 19

`WWSWWWS` (class in `aruudy.poetry.foot`), 19

`WWWWSWWS` (class in `aruudy.poetry.foot`), 19

Z

`ZuhafIlla` (class in `aruudy.poetry.foot`), 19