
IEMML Library Documentation

Release 0.1.0

Hadrien Titeux

Mar 08, 2021

Table of Content

1	An Overview of the IEML metalanguage	3
2	An introduction to the IEML Library	5
3	API Reference	7
3.1	Dictionary reference	7
3.1.1	Script Structure	7
3.1.2	Dictionary Structure	9
3.1.3	Script Parser	9
3.2	USL Object Reference	11
3.2.1	USL Object Structure	11
3.2.2	Table Structure	16
3.2.3	USL Parser	17
3.2.4	Path Parser	18
3.2.5	Literals	22
3.3	Utils	23
3.4	Database reference	26
	Python Module Index	31
	Index	33

This is the home of the documentation for the Python implementation of the IEML metalanguage. You should find here

- a brief intro to the theory behind the IEML metalanguage
- examples on how to use our implementation
- an API reference

Bear in mind that the project is still under heavy development (and conception). You should use this documentation as a way to get a better understanding of the IEML project, and a basic overview of our Python implementation's usage.

CHAPTER 1

An Overview of the IEMML metalanguage

CHAPTER 2

An introduction to the IEML Library

3.1 Dictionary reference

3.1.1 Script Structure

class `ieuml.dictionary.script.script.AdditiveScript` (*children=None, character=None*)
 Bases: `ieuml.dictionary.script.script.Script`

Represent an addition of same layer scripts.

__init__ (*children=None, character=None*)
 Initialize self. See `help(type(self))` for accurate signature.

__module__ = `'ieuml.dictionary.script.script'`

class `ieuml.dictionary.script.script.MultiplicativeScript` (*substance=None, attribute=None, mode=None, children=None, character=None*)
 Bases: `ieuml.dictionary.script.script.Script`

Represent a multiplication of three scripts of the same layer.

__init__ (*substance=None, attribute=None, mode=None, children=None, character=None*)
 Initialize self. See `help(type(self))` for accurate signature.

__module__ = `'ieuml.dictionary.script.script'`

class `ieuml.dictionary.script.script.NullScript` (*layer*)
 Bases: `ieuml.dictionary.script.script.Script`

__init__ (*layer*)
 Initialize self. See `help(type(self))` for accurate signature.

__iter__ ()
 Enables the syntactic sugar of iterating directly on an element without accessing “children”

```

__module__ = 'ieml.dictionary.script.script'
class ieml.dictionary.script.script.Script (children=None, character=None, *args,
                                             **kwargs)
    Bases: ieml.common.TreeStructure, ieml.common.DecoratedComponent
    A parser is defined by a character (PRIMITIVES, REMARKABLE_ADDITION OR REMARK-
    ABLE_MULTIPLICATION) or a list of parser children. All the element in the children list must be an Ad-
    ditveScript or a MultiplicativeScript.

    __add__ (other)

    __contains__ (item)

    __eq__ (other)
        Return self==value.

    __getnewargs_ex__ ()

    __hash__ ()
        Since the IEML string for a script is its definition, it can be used as a hash

    __init__ (children=None, character=None, *args, **kwargs)
        Initialize self. See help(type(self)) for accurate signature.

    __len__ ()

    __lt__ (other)
        Return self<value.

    __module__ = 'ieml.dictionary.script.script'

    static __new__ (cls, *args, **kwargs)
        Need this to pickle scripts, the pickler use __hash__ method before unpickling the object attribute. Then
        need to pass the _str.

    cells

    check ()

    headers

    is_singular

    iter_structure ()

    iter_structure_path (flexion=False)

    iter_structure_path_by_script_ss (flexion=False)

    singular_sequences

    singular_sequences_set

    tables_script

ieml.dictionary.script.operator.add (scripts: List[ieml.dictionary.script.script.Script])
ieml.dictionary.script.operator.m (substance, attribute=None, mode=None)
ieml.dictionary.script.operator.script (arg, promote=False, factorize=False)
ieml.dictionary.script.tools.factor (sequences)
ieml.dictionary.script.tools.factorize (script: Union[ieml.dictionary.script.script.Script,
List[ieml.dictionary.script.script.Script]], promote:
bool = True) → ieml.dictionary.script.script.Script

```

Parameters

- **script** – The Script or list of Script to factorize
- **promote** – If script is a list, promote all Script to the layer max(sc.layer for sc in scripts)

Returns the factorized script

`ieml.dictionary.script.tools.pack_factorisation` (*facto_list*)

Parameters **facto_list** – list of script or tuple of factorisation

Returns

`ieml.dictionary.script.tools.promote` (*script: ieml.dictionary.script.script.Script, layer: int*)
Promote script to layer by multiplying it with null scripts (E:) :param script: :param layer: :return:

3.1.2 Dictionary Structure

class `ieml.dictionary.dictionary.Dictionary` (*paradigms, structure*)

Bases: `object`

`__contains__` (*item*)

`__dict__` = `mappingproxy({'__module__': 'ieml.dictionary.dictionary', '__init__': <fun`

`__getitem__` (*item*)

`__init__` (*paradigms, structure*)

Initialize self. See help(type(self)) for accurate signature.

`__len__` ()

`__module__` = 'ieml.dictionary.dictionary'

`__weakref__`

list of weak references to the object (if defined)

3.1.3 Script Parser

`ieml.dictionary.script.parser.lexer.get_script_lexer` (*module=None*)

class `ieml.dictionary.script.parser.parser.ScriptParser`

Bases: `object`

`__dict__` = `mappingproxy({'__module__': 'ieml.dictionary.script.parser.parser', 'token`

`__init__` ()

Initialize self. See help(type(self)) for accurate signature.

`__module__` = 'ieml.dictionary.script.parser.parser'

`__weakref__`

list of weak references to the object (if defined)

`lock` = `<unlocked _thread.lock object>`

`p_additive_script_lvl_0` (*p*)

additive_script_lvl_0 : sum_lvl_0

`p_additive_script_lvl_1` (*p*)

additive_script_lvl_1 : sum_lvl_1

```

p_additive_script_lvl_2 (p)
    additive_script_lvl_2 : sum_lvl_2

p_additive_script_lvl_3 (p)
    additive_script_lvl_3 : sum_lvl_3

p_additive_script_lvl_4 (p)
    additive_script_lvl_4 : sum_lvl_4

p_additive_script_lvl_5 (p)
    additive_script_lvl_5 : sum_lvl_5

p_additive_script_lvl_6 (p)
    additive_script_lvl_6 : sum_lvl_6

p_error (p)

p_script_lvl_0 (p)
    script_lvl_0 : PRIMITIVE LAYER0_MARK | REMARKABLE_ADDITION LAYER0_MARK

p_script_lvl_1 (p)
    script_lvl_1 : additive_script_lvl_0 LAYER1_MARK | additive_script_lvl_0 additive_script_lvl_0
    LAYER1_MARK | additive_script_lvl_0 additive_script_lvl_0 additive_script_lvl_0 LAYER1_MARK |
    REMARKABLE_MULTIPLICATION LAYER1_MARK

p_script_lvl_2 (p)
    script_lvl_2 : sum_lvl_1 LAYER2_MARK | sum_lvl_1 sum_lvl_1 LAYER2_MARK | sum_lvl_1
    sum_lvl_1 sum_lvl_1 LAYER2_MARK

p_script_lvl_3 (p)
    script_lvl_3 : sum_lvl_2 LAYER3_MARK | sum_lvl_2 sum_lvl_2 LAYER3_MARK | sum_lvl_2
    sum_lvl_2 sum_lvl_2 LAYER3_MARK

p_script_lvl_4 (p)
    script_lvl_4 : sum_lvl_3 LAYER4_MARK | sum_lvl_3 sum_lvl_3 LAYER4_MARK | sum_lvl_3
    sum_lvl_3 sum_lvl_3 LAYER4_MARK

p_script_lvl_5 (p)
    script_lvl_5 : sum_lvl_4 LAYER5_MARK | sum_lvl_4 sum_lvl_4 LAYER5_MARK | sum_lvl_4
    sum_lvl_4 sum_lvl_4 LAYER5_MARK

p_script_lvl_6 (p)
    script_lvl_6 : sum_lvl_5 LAYER6_MARK | sum_lvl_5 sum_lvl_5 LAYER6_MARK | sum_lvl_5
    sum_lvl_5 sum_lvl_5 LAYER6_MARK

p_sum_lvl_0 (p)
    sum_lvl_0 : script_lvl_0 | script_lvl_0 PLUS sum_lvl_0

p_sum_lvl_1 (p)
    sum_lvl_1 : script_lvl_1 | script_lvl_1 PLUS sum_lvl_1

p_sum_lvl_2 (p)
    sum_lvl_2 : script_lvl_2 | script_lvl_2 PLUS sum_lvl_2

p_sum_lvl_3 (p)
    sum_lvl_3 : script_lvl_3 | script_lvl_3 PLUS sum_lvl_3

p_sum_lvl_4 (p)
    sum_lvl_4 : script_lvl_4 | script_lvl_4 PLUS sum_lvl_4

p_sum_lvl_5 (p)
    sum_lvl_5 : script_lvl_5 | script_lvl_5 PLUS sum_lvl_5

```

```

p_sum_lvl_6(p)
    sum_lvl_6 : script_lvl_6 | script_lvl_6 PLUS sum_lvl_6

p_term(p)
    term : script_lvl_0 | additive_script_lvl_0 | script_lvl_1 | additive_script_lvl_1 | script_lvl_2 | additive_script_lvl_2 | script_lvl_3 | additive_script_lvl_3 | script_lvl_4 | additive_script_lvl_4 | script_lvl_5 | additive_script_lvl_5 | script_lvl_6 | additive_script_lvl_6

t_add_rules()

t_parse

tokens = ('PLUS', 'LAYER0_MARK', 'LAYER1_MARK', 'LAYER2_MARK', 'LAYER3_MARK', 'LAYER4_MARK')

```

3.2 USL Object Reference

```

ieml.usl.get_index(s, dic)

ieml.usl.int2base(i, max=30, characters=['0', '1', '2', '3', '4', '5', '6', '7', '8', '9', 'A', 'B', 'C', 'D', 'E', 'F', 'G', 'H', 'I', 'J', 'K', 'L', 'M', 'N', 'O', 'P', 'Q', 'R', 'S', 'T', 'U', 'V', 'W', 'X', 'Y', 'Z', 'a', 'b', 'c', 'd', 'e', 'f', 'g', 'h', 'i', 'j', 'k', 'l', 'm', 'n', 'o', 'p', 'q', 'r', 's', 't', 'u', 'v', 'w', 'x', 'y', 'z'])

```

3.2.1 USL Object Structure

```

class ieml.usl.usl.USL(*args, **kwargs)
    Bases: ieml.common.DecoratedComponent

    __bool__()

    __contains__(item)

    __eq__(other)
        Return self==value.

    __hash__()
        Since the IEML string for a script is its definition, it can be used as a hash

    __init__(*args, **kwargs)
        Initialize self. See help(type(self)) for accurate signature.

    __len__()

    __lt__(other)
        Return self<value.

    __module__ = 'ieml.usl.usl'

    __str__()
        Return str(self).

    cardinal

    check()

    do_lt(other)

    empty

    is_singular

```

```

iter_structure()
iter_structure_path(flexion=False) → Iterable[Tuple[UslPath, USL]]
iter_structure_path_by_script_ss(flexion=False) → Iterable[Tuple[UslPath,
    ieml.dictionary.script.script.Script]]
iter_structure_path_by_type(_type=<class 'ieml.dictionary.script.script.Script'>, flex-
    ion=False)

morphemes
singular_sequences
singular_sequences_set
syntactic_level = 0
ieml.usl.usl.usl(arg: Union[str, ieml.dictionary.script.script.Script, ieml.usl.usl.USL, Iter-
    able[Tuple[UslPath, Union[ieml.usl.usl.USL, ieml.dictionary.script.script.Script]]]])
    → ieml.usl.usl.USL

Cast argument to an USL type, depending on the argument type.

    • If argument is a string, it is parsed by ieml.usl.parser.IEMLParser.parse
    • if argument is a ieml.dictionary.Script, the returned object is a ieml.usl.polymorpheme.PolyMorpheme
      with the argument as the constant.
    • if argument is an ieml.usl.usl.USL, the argument is returned
    • if argument is a list of (ieml.usl.decoration.path.UslPath, ieml.usl.usl.USL)

Parameters arg (Union[str, Script, USL, Iterable[Tuple['UslPath',
    Union[USL, Script]]]])-

Returns an ieml.usl.usl.USL

class ieml.usl.polymorpheme.PolyMorpheme(constant: List[ieml.dictionary.script.script.Script]
    = (), groups=())
    Bases: ieml.usl.usl.USL
    __init__(constant: List[ieml.dictionary.script.script.Script] = (), groups=())
        Initialize self. See help(type(self)) for accurate signature.
    __module__ = 'ieml.usl.polymorpheme'
    check()
    do_lt(other)
    empty
    iter_structure()
    iter_structure_path(flexion=False)
    morphemes
    syntactic_level = 1
ieml.usl.polymorpheme.check_polymorpheme(ms)
ieml.usl.polymorpheme.compute_PM_singular_sequences(constants, groups)

```



```
class ieml.usl.lexeme.Lexeme (pm_flexion: ieml.usl.polymorpheme.PolyMorpheme, pm_content:
                                ieml.usl.polymorpheme.PolyMorpheme)
```

Bases: *ieml.usl.usl.USL*

A lexeme without the PA of the position on the tree (position independant lexeme)

```
__init__ (pm_flexion: ieml.usl.polymorpheme.PolyMorpheme, pm_content:
            ieml.usl.polymorpheme.PolyMorpheme)
    Initialize self. See help(type(self)) for accurate signature.
```

```
__module__ = 'ieml.usl.lexeme'
```

```
check()
```

```
do_lt (other)
```

```
empty
```

```
iter_structure()
```

```
iter_structure_path (flexion=False)
```

```
morphemes
```

```
syntactic_level = 2
```

```
ieml.usl.lexeme.check_lexeme (lexeme, sfun=None)
```

```
class ieml.usl.syntagmatic_function.DependantQualitySyntagmaticFunction (actor:
```

Any,

de-

pen-

dant:

Union[DependantQualitySyntagmaticFunction,

ieml.usl.syntagmatic_function.JunctionSyntagmaticFunction]

=

None,

in-

de-

pen-

dant:

Union[ieml.usl.syntagmatic_function.JunctionSyntagmaticFunction,

ieml.usl.syntagmatic_function.IndependentQualitySyntagmaticFunction]

=

None,

***kwargs)*

Bases: *ieml.usl.syntagmatic_function.SyntagmaticFunction*

```
__init__ (actor: Any, dependant: Union[DependantQualitySyntagmaticFunction,
            ieml.usl.syntagmatic_function.JunctionSyntagmaticFunction] = None, indepen-
            dant: Union[ieml.usl.syntagmatic_function.IndependentQualitySyntagmaticFunction,
            ieml.usl.syntagmatic_function.JunctionSyntagmaticFunction] = None, **kwargs)
    Initialize self. See help(type(self)) for accurate signature.
```

```
__module__ = 'ieml.usl.syntagmatic_function'
```

```
check (X, check_X, sfun_type)
```

```
class ieml.usl.syntagmatic_function.IndependentQualitySyntagmaticFunction (actor:
```

Any)

Bases: *ieml.usl.syntagmatic_function.SyntagmaticFunction*

```

    __init__(actor: Any)
        Initialize self. See help(type(self)) for accurate signature.

    __module__ = 'ieml.usl.syntagmatic_function'

class ieml.usl.syntagmatic_function.JunctionSyntagmaticFunction (junction_link:
                                                                    ieml.dictionary.script.script.Script,
                                                                    children:
                                                                    List[ieml.usl.syntagmatic_function.SyntagmaticFunction]

    Bases: ieml.usl.syntagmatic_function.SyntagmaticFunction

    __init__(junction_link: ieml.dictionary.script.script.Script, children:
                List[ieml.usl.syntagmatic_function.SyntagmaticFunction])
        Initialize self. See help(type(self)) for accurate signature.

    __module__ = 'ieml.usl.syntagmatic_function'

    check(X: Type[CT_co], check_X, sfun_type)

class ieml.usl.syntagmatic_function.ProcessSyntagmaticFunction (actor: Any,
                                                                    valence:
                                                                    ieml.dictionary.script.script.Script,
                                                                    actants:
                                                                    Dict[ieml.dictionary.script.script.Script,
                                                                    ieml.usl.syntagmatic_function.DependantQualitySyntagmaticFunction]

    Bases: ieml.usl.syntagmatic_function.SyntagmaticFunction

    __init__(actor: Any, valence: ieml.dictionary.script.script.Script, actants:
                Dict[ieml.dictionary.script.script.Script, ieml.usl.syntagmatic_function.DependantQualitySyntagmaticFunction])
        Initialize self. See help(type(self)) for accurate signature.

    __module__ = 'ieml.usl.syntagmatic_function'

    check(X: Type[CT_co], check_X, sfun_type)

class ieml.usl.syntagmatic_function.SyntagmaticFunction (actor: Any, _actors:
                                                            Dict[List[ieml.dictionary.script.script.Script],
                                                            SyntagmaticFunction])

    Bases: object

    __dict__ = mappingproxy({'__module__': 'ieml.usl.syntagmatic_function', '__init__':
    __eq__(other)
        Return self==value.

    __hash__ = None

    __init__(actor: Any, _actors: Dict[List[ieml.dictionary.script.script.Script], SyntagmaticFunction])
        Initialize self. See help(type(self)) for accurate signature.

    __lt__(other)
        Return self<value.

    __module__ = 'ieml.usl.syntagmatic_function'

    __weakref__
        list of weak references to the object (if defined)

    as_list(context_type) → List[Tuple[List[ieml.dictionary.script.script.Script], Any]]

    check(X: Type[CT_co], check_X, sfun_type)

    empty

```

```

static from_list (l: List[Tuple[List[ieml.dictionary.script.script.Script], Any]]) →
    Tuple[Type[ieml.usl.syntagmatic_function.SyntagmaticFunction],
    ieml.usl.syntagmatic_function.SyntagmaticFunction]

get (role: ieml.usl.syntagmatic_function.SyntagmaticRole, ignore_prefix=False, ignore_process_valence=False) → Any

static get_context_role_prefix (context)

get_paradigm (role: ieml.usl.syntagmatic_function.SyntagmaticRole) → List[Any]

get_role_expansion (role: ieml.usl.syntagmatic_function.SyntagmaticRole, ignore_prefix=())

iter_structure ()

iter_structure_path (context, focus_role=None)

render_with_context (role: ieml.usl.syntagmatic_function.SyntagmaticRole = None, context=None)

role_in (tgt: ieml.usl.syntagmatic_function.SyntagmaticFunction)
    return the role of the tgt syntagmatic function occupy in self.

role_is_junction (role: ieml.usl.syntagmatic_function.SyntagmaticRole = None)

singular_sequences (context_type)

class ieml.usl.syntagmatic_function.SyntagmaticRole (constant:
    List[ieml.dictionary.script.script.Script]
    = ())

Bases: object

__dict__ = mappingproxy({'__module__': 'ieml.usl.syntagmatic_function', '__init__':
__eq__ (other)
    Return self==value.

__hash__ ()
    Since the IEML string for a script is its definition, it can be used as a hash

__init__ (constant: List[ieml.dictionary.script.script.Script] = ())
    Initialize self. See help(type(self)) for accurate signature.

__lt__ (other)
    Return self<value.

__module__ = 'ieml.usl.syntagmatic_function'

__str__ ()
    Return str(self).

__weakref__
    list of weak references to the object (if defined)

is_junction_prefix (role: ieml.usl.syntagmatic_function.SyntagmaticRole)
    if self is the junction prefix of the role

class ieml.usl.word.Word (syntagmatic_fun: ieml.usl.syntagmatic_function.SyntagmaticFunction,
    role: ieml.usl.syntagmatic_function.SyntagmaticRole, context_type:
    Type[ieml.usl.syntagmatic_function.SyntagmaticFunction])

Bases: ieml.usl.usl.USL

__init__ (syntagmatic_fun: ieml.usl.syntagmatic_function.SyntagmaticFunction,
    role: ieml.usl.syntagmatic_function.SyntagmaticRole, context_type:
    Type[ieml.usl.syntagmatic_function.SyntagmaticFunction])
    Initialize self. See help(type(self)) for accurate signature.
    
```

```

__module__ = 'ieml.usl.word'
check()
do_lt(other)
empty
iter_structure()
iter_structure_path(flexion=False)
morphemes
syntactic_level = 3
ieml.usl.word.check_word(w: ieml.usl.word.Word)
ieml.usl.word.simplify_word(w: ieml.usl.word.Word) → ieml.usl.word.Word
    remove empty leaves

```

3.2.2 Table Structure

```

class ieml.usl.table.UslTable2D(usl: ieml.usl.usl.USL, columns: ieml.usl.decoration.path.UslPath, rows: ieml.usl.decoration.path.UslPath = None)
    Bases: object
    __dict__ = mappingproxy({'__module__': 'ieml.usl.table', '__init__': <function UslTable2D.__init__ at 0x...>})
    __init__(usl: ieml.usl.usl.USL, columns: ieml.usl.decoration.path.UslPath, rows: ieml.usl.decoration.path.UslPath = None)
        Initialize self. See help(type(self)) for accurate signature.
    __module__ = 'ieml.usl.table'
    __weakref__
        list of weak references to the object (if defined)
    cells
    column_paths_constant
    column_paths_variation
    columns
    constant_paths
    row_paths_constant
    row_paths_variation
    rows
ieml.usl.table.enumerate_partitions(usl: ieml.usl.usl.USL)
class ieml.usl.variation.PolyMorphemeVariation(items, multiplicity)
    Bases: ieml.usl.variation.Variation
    __init__(items, multiplicity)
        Initialize self. See help(type(self)) for accurate signature.
    __module__ = 'ieml.usl.variation'
    check()

```

```

do_lt (other)

empty

iter_structure ()

iter_structure_path (flexion=False)

morphemes

class ieml.usl.variation.Variation (*args, **kwargs)
    Bases: ieml.usl.usl.USL
    __module__ = 'ieml.usl.variation'

```

3.2.3 USL Parser

```

ieml.usl.parser.lexer.get_lexer (module=None)

class ieml.usl.parser.parser.IEMLParser (dictionary=None)
    Bases: object
    __dict__ = mappingproxy({'__module__': 'ieml.usl.parser.parser', 'tokens': ('MORPHEM
    __init__ (dictionary=None)
        Initialize self. See help(type(self)) for accurate signature.
    __module__ = 'ieml.usl.parser.parser'
    __weakref__
        list of weak references to the object (if defined)
    lock = <unlocked _thread.lock object>
    p_decoration (p)
        decoration : LBRACKET USL_PATH DECORATION_VALUE RBRACKET
    p_decoration_list (p)
        decoration_list : decoration_list decoration | decoration
    p_error (p)
    p_group (p)
        group : GROUP_MULTIPPLICITY LPAREN morpheme_sum RPAREN
    p_group_list (p)
        group_list : group group_list | group
    p_ieml_proposition (p)
        proposition : morpheme | usl | instanced_usl
    p_instanced_usl (p)
        instanced_usl : usl decoration_list
    p_lexeme (p)
        lexeme : LPAREN poly_morpheme RPAREN LPAREN poly_morpheme RPAREN LPAREN
        poly_morpheme RPAREN | LPAREN RPAREN LPAREN poly_morpheme RPAREN LPAREN
        poly_morpheme RPAREN | LPAREN RPAREN LPAREN RPAREN LPAREN poly_morpheme RPAREN
        | LPAREN poly_morpheme RPAREN LPAREN poly_morpheme RPAREN | LPAREN RPAREN LPAREN
        poly_morpheme RPAREN | LPAREN poly_morpheme RPAREN | LPAREN RPAREN
    p_lexeme_list (p)
        lexeme_list : lexeme_list RCHEVRON EXCLAMATION_MARK positioned_lexeme | lexeme_list
        RCHEVRON positioned_lexeme | EXCLAMATION_MARK positioned_lexeme | positioned_lexeme

```

```

p_morpheme (p)
    morpheme : MORPHEME

p_morpheme_sum (p)
    morpheme_sum : morpheme_sum morpheme | morpheme

p_poly_morpheme (p)
    poly_morpheme : morpheme_sum group_list | morpheme_sum | group_list

p_positioned_lexeme (p)
    positioned_lexeme : morpheme_sum lexeme | lexeme

p_usl (p)
    usl : poly_morpheme | lexeme | word

p_word (p)
    word : LBRACKET OLD_MORPHEME_GRAMMATICAL_CLASS lexeme_list RBRACKET |
    LBRACKET lexeme_list RBRACKET

parse (s, factorize_script=False)
    Parses the input string, and returns a reference to the created AST's root

tokens = ('MORPHEME', 'OLD_MORPHEME_GRAMMATICAL_CLASS', 'LPAREN', 'RPAREN', 'RCHEVRON')

class ieml.usl.parser.parser.IEMLParserSingleton
    Bases: type

    __call__ (*args, **kwargs)
        Call self as a function.

    __module__ = 'ieml.usl.parser.parser'

```

3.2.4 Path Parser

```

exception ieml.usl.decoration.path.DeferenceError
    Bases: KeyError

    __module__ = 'ieml.usl.decoration.path'

    __weakref__
        list of weak references to the object (if defined)

class ieml.usl.decoration.path.FlexionPath (morpheme, child=None)
    Bases: ieml.usl.decoration.path.UslPath

    USL_TYPE
        alias of ieml.usl.polymorpheme.PolyMorpheme

    __init__ (morpheme, child=None)
        Initialize self. See help(type(self)) for accurate signature.

    __module__ = 'ieml.usl.decoration.path'

    as_constant (u=None)

    classmethod build_usl_from_path_to_node (path_to_node)

    clone (use_child=False, child=None)

    without_morpheme ()

class ieml.usl.decoration.path.GroupIndex
    Bases: ieml.common.OrderedEnum

```

An enumeration.

CONSTANT = -1

GROUP_0 = 0

GROUP_1 = 1

GROUP_2 = 2

__module__ = 'ieml.usl.decoration.path'

class ieml.usl.decoration.path.LexemeIndex

Bases: *ieml.common.OrderedEnum*

An enumeration.

CONTENT = 0

FLEXION = 1

__module__ = 'ieml.usl.decoration.path'

class ieml.usl.decoration.path.LexemePath(index: *ieml.usl.decoration.path.LexemeIndex*,
child=None)

Bases: *ieml.usl.decoration.path.UslPath*

USL_TYPE

alias of *ieml.usl.lexeme.Lexeme*

__init__(index: *ieml.usl.decoration.path.LexemeIndex*, child=None)

Initialize self. See help(type(self)) for accurate signature.

__module__ = 'ieml.usl.decoration.path'

as_constant(u=None)

classmethod build_usl_from_path_to_node(path_to_node)

clone(use_child=False, child=None)

without_morpheme()

class ieml.usl.decoration.path.PolymorphemePath(group_idx:
ieml.usl.decoration.path.GroupIndex,
morpheme:
ieml.dictionary.script.script.Script
= None, multiplicity=None,
child=None)

Bases: *ieml.usl.decoration.path.UslPath*

USL_TYPE

alias of *ieml.usl.polymorpheme.PolyMorpheme*

__init__(group_idx: *ieml.usl.decoration.path.GroupIndex*, morpheme:

ieml.dictionary.script.script.Script = None, multiplicity=None, child=None)

Initialize self. See help(type(self)) for accurate signature.

__module__ = 'ieml.usl.decoration.path'

as_constant(u=None)

classmethod build_usl_from_path_to_node(path_to_node)

path_to_node: dict PolymorphemePath -> Script # TODO handle multiplicity :param path_to_node: :return:

clone(use_child=False, child=None)

```

    without_morpheme ()

class ieml.usl.decoration.path.RolePath (role, has_focus=False, child=None)
    Bases: ieml.usl.decoration.path.UslPath

    USL_TYPE
        alias of ieml.usl.word.Word

    __init__ (role, has_focus=False, child=None)
        Initialize self. See help(type(self)) for accurate signature.

    __module__ = 'ieml.usl.decoration.path'

    as_constant (u=None)

    classmethod build_usl_from_path_to_node (path_to_node)

    clone (use_child=False, child=None)

    without_morpheme ()

class ieml.usl.decoration.path.UslPath (child=None)
    Bases: object

    USL_TYPE
        alias of ieml.usl.usl.USL

    __dict__ = mappingproxy({'__module__': 'ieml.usl.decoration.path', 'USL_TYPE': <class
    __eq__ (other)
        Return self==value.

    __hash__ ()
        Return hash(self).

    __init__ (child=None)
        Initialize self. See help(type(self)) for accurate signature.

    __lt__ (other)
        Return self<value.

    __module__ = 'ieml.usl.decoration.path'

    __str__ ()
        Return str(self).

    __weakref__
        list of weak references to the object (if defined)

    as_constant (u=None)

    classmethod build_usl_from_path_to_node (path_to_node)

    clone (use_child=False, child=None)

    concat (suffix: ieml.usl.decoration.path.UslPath, force: bool = False) →
        ieml.usl.decoration.path.UslPath

    contained (usl)

    deference (usl: ieml.usl.usl.USL) → ieml.usl.usl.USL

    classmethod from_string (string)

    has_prefix (prefix: ieml.usl.decoration.path.UslPath)

    is_constant_path

```



```

no_child_clone()

remove_prefix(prefix: ieml.usl.decoration.path.UslPath)

split_tail()

tail

without_morpheme()

ieml.usl.decoration.path.path(string) → ieml.usl.decoration.path.UslPath
ieml.usl.decoration.path.usl_from_path_values(paths_values)
ieml.usl.decoration.parser.lexer.get_lexer(module=None)
class ieml.usl.decoration.parser.parser.PathParser
    Bases: object
    __dict__ = mappingproxy({'__module__': 'ieml.usl.decoration.parser.parser', 'tokens':
    __init__()
        Initialize self. See help(type(self)) for accurate signature.
    __module__ = 'ieml.usl.decoration.parser.parser'
    __weakref__
        list of weak references to the object (if defined)
    lock = <unlocked _thread.lock object>
    p_error(p)
    p_flexion_path(p)
        flexion_path : MORPHEME
    p_lexeme_path(p)
        lexeme_path : LEXEME_POSITION | LEXEME_POSITION SEPARATOR polymorpheme_path | LEX-
        EME_POSITION SEPARATOR flexion_path
    p_path(p)
        path : SEPARATOR | SEPARATOR role_path | SEPARATOR lexeme_path | SEPARATOR flexion_path |
        SEPARATOR polymorpheme_path
    p_polymorpheme_path(p)
        polymorpheme_path : POLYMORPHEME_POSITION | POLYMORPHEME_POSITION MULTIPLIC-
        ITY | POLYMORPHEME_POSITION SEPARATOR MORPHEME | POLYMORPHEME_POSITION
        MULTIPLICITY SEPARATOR MORPHEME
    p_role_path(p)
        role_path : ROLE_TOKEN SEPARATOR role_path_list | ROLE_TOKEN SEPARATOR EXCLAMA-
        TION_MARK role_path_list | ROLE_TOKEN SEPARATOR role_path_list SEPARATOR lexeme_path |
        ROLE_TOKEN SEPARATOR EXCLAMATION_MARK role_path_list SEPARATOR lexeme_path
    p_role_path_list(p)
        role_path_list : role_path_list MORPHEME | MORPHEME | role_path_list ROLE_NAME |
        ROLE_NAME
    parse(s)
    tokens = ('SEPARATOR', 'ROLE_TOKEN', 'ROLE_NAME', 'ROLE_MORPHEME', 'LEXEME_POSITION',

```

3.2.5 Literals

```
class ieml.usl.decoration.instance.Decoration (path: ieml.usl.decoration.path.UslPath,  
                                              value)
```

Bases: `object`

```
__dict__ = mappingproxy({'__module__': 'ieml.usl.decoration.instance', '__init__': <
```

```
__eq__ (other)
```

Return self==value.

```
__hash__ = None
```

```
__init__ (path: ieml.usl.decoration.path.UslPath, value)
```

Initialize self. See help(type(self)) for accurate signature.

```
__lt__ (other)
```

Return self<value.

```
__module__ = 'ieml.usl.decoration.instance'
```

```
__str__ ()
```

Return str(self).

```
__weakref__
```

list of weak references to the object (if defined)

```
apply (u: ieml.common.DecoratedComponent)
```

```
class ieml.usl.decoration.instance.InstancedUSL (u: ieml.usl.usl.USL, decorations:  
                                                  List[ieml.usl.decoration.instance.Decoration])
```

Bases: `ieml.usl.usl.USL`

```
__init__ (u: ieml.usl.usl.USL, decorations: List[ieml.usl.decoration.instance.Decoration])
```

Initialize self. See help(type(self)) for accurate signature.

```
__module__ = 'ieml.usl.decoration.instance'
```

```
__str__ ()
```

Return str(self).

```
check ()
```

```
do_lt (other)
```

```
empty
```

```
static from_usl (u: ieml.usl.usl.USL)
```

```
iter_structure ()
```

```
iter_structure_path (flexion=False)
```

```
static list_decorations (u: ieml.usl.usl.USL, flexion=False)
```

```
morphemes
```

```
syntactic_level = 10
```

```
class ieml.usl.decoration.instance.LiteralContext
```

Bases: `object`

```
__dict__ = mappingproxy({'__module__': 'ieml.usl.decoration.instance', '__init__': <
```

```
__enter__ ()
```

```
__exit__ (exc_type, exc_val, exc_tb)
```

```

__init__()
    Initialize self. See help(type(self)) for accurate signature.

__module__ = 'ieml.usl.decoration.instance'

__weakref__
    list of weak references to the object (if defined)

push(u)

ieml.usl.decoration.instance.literal_context()

```

3.3 Utils

```

ieml.usl.constants.assert_(cond, message)

ieml.usl.constants.assert_all_in(l: List[ieml.dictionary.script.script.Script], _set:
                                Set[ieml.dictionary.script.script.Script], name_l)

ieml.usl.constants.assert_atmost_one_from(l: List[ieml.dictionary.script.script.Script],
                                          _set: Set[ieml.dictionary.script.script.Script],
                                          name_l, name_set)

ieml.usl.constants.assert_no_one_from(l: List[ieml.dictionary.script.script.Script], _set:
                                      Set[ieml.dictionary.script.script.Script], name_l,
                                      name_set)

ieml.usl.constants.assert_only_one_from(l: List[ieml.dictionary.script.script.Script], _set:
                                       Set[ieml.dictionary.script.script.Script], name_l,
                                       name_set) → ieml.dictionary.script.script.Script

ieml.usl.constants.check_address_script(l: List[ieml.dictionary.script.script.Script],
                                         sfun_type)

ieml.usl.constants.check_flexion_actant_scripts(l: List[ieml.dictionary.script.script.Script],
                                                sfun=None)

ieml.usl.constants.check_flexion_process_scripts(l: List[ieml.dictionary.script.script.Script],
                                                  sfun=None)

ieml.usl.constants.check_flexion_quality(l: List[ieml.dictionary.script.script.Script],
                                         sfun=None)

ieml.usl.constants.check_lexeme_scripts(l_pf: List[ieml.dictionary.script.script.Script],
                                         l_pc: List[ieml.dictionary.script.script.Script],
                                         sfun=None)

ieml.usl.constants.class_from_address(address)

class ieml.common.DecoratedComponent(*args, **kwargs)
    Bases: object
    __dict__ = mappingproxy({'__module__': 'ieml.common', '__init__': <function DecoratedComponent.__init__>})
    __init__(*args, **kwargs)
        Initialize self. See help(type(self)) for accurate signature.
    __module__ = 'ieml.common'
    __weakref__
        list of weak references to the object (if defined)
    clear_literal()

```

```
get_literal()
```

```
set_literal(value)
```

```
class ieml.common.FolderWatcherCache(db_path: str, pattern: str, cache_folder: str, name: str)
```

Bases: `object`

```
__dict__ = mappingproxy({'__module__': 'ieml.common', '__init__': <function FolderW
```

```
__init__(db_path: str, pattern: str, cache_folder: str, name: str)
```

Cache that check if *folder* content has changed. Compute a hash of the files in the folder and get pruned if the content of this folder change.

Parameters

- **folder** – the folder to watch
- **cache_folder** – the folder to put the cache file

```
__module__ = 'ieml.common'
```

```
__weakref__
```

list of weak references to the object (if defined)

```
cache_file
```

Returns The cache file absolute path

```
get() → object
```

Unpickle and return the object stored in the cache file. :return: the stored object

```
is_pruned() → bool
```

Return True if the watched folder content has changed. :return: if the folder content changed

```
update(obj) → None
```

Update the cache content, remove old cache files from the cache directory.

Parameters *obj* – the object to pickle in the cache

Returns None

```
class ieml.common.LastUpdatedOrderedDict
```

Bases: `collections.OrderedDict`

Store items in the order the keys were last added

```
__module__ = 'ieml.common'
```

```
__setitem__(key, value, **kwargs)
```

Set self[key] to value.

```
class ieml.common.OrderedEnum
```

Bases: `enum.Enum`

An enumeration.

```
__ge__(other)
```

Return self>=value.

```
__gt__(other)
```

Return self>value.

```
__le__(other)
```

Return self<=value.

```

    __lt__(other)
        Return self<value.

    __module__ = 'ieml.common'

class ieml.common.Singleton
    Bases: type

    __call__(*args, **kwargs)
        Call self as a function.

    __module__ = 'ieml.common'

class ieml.common.TreeStructure(*args, **kwargs)
    Bases: object

    __dict__ = mappingproxy({'__module__': 'ieml.common', '__init__': <function TreeStru
    __eq__(other)
        Return self==value.

    __hash__()
        Since the IEMML string for any proposition AST is supposed to be unique, it can be used as a hash

    __init__(*args, **kwargs)
        Initialize self. See help(type(self)) for accurate signature.

    __iter__()
        Enables the syntactic sugar of iterating directly on an element without accessing "children"

    __module__ = 'ieml.common'

    __ne__(other)
        Return self!=value.

    __str__()
        Return str(self).

    __weakref__
        list of weak references to the object (if defined)

    tree_iter()

ieml.common.cache_results_watch_files(path, name)

class ieml.common.cached_property(factory)
    Bases: object

    __dict__ = mappingproxy({'__module__': 'ieml.common', '__init__': <function cached_p
    __get__(instance, owner)

    __init__(factory)
        Initialize self. See help(type(self)) for accurate signature.

    __module__ = 'ieml.common'

    __weakref__
        list of weak references to the object (if defined)

ieml.common.fullname(cls)

ieml.common.monitor_decorator(name)

class ieml.constants.DescriptorsType
    Bases: enum.Enum

```

An enumeration.

```
COMMENTS = 'comments'
REFERENTIAL = 'rdf_repositories'
TAGS = 'tags'
TRANSLATIONS = 'translations'
__module__ = 'ieml.constants'
```

```
class ieml.constants.Languages
```

Bases: `enum.Enum`

The language currently supported by the IEML database

```
EN = 'en'
FR = 'fr'
__module__ = 'ieml.constants'
```

```
ieml.constants.get_iemlldb_folder(name)
```

3.4 Database reference

```
class ieml.ieml_database.descriptors.Descriptors(df)
```

Bases: `object`

```
__dict__ = mappingproxy({'__module__': 'ieml.ieml_database.descriptors', '__init__':
__init__(df)
```

Initialize self. See help(type(self)) for accurate signature.

```
__module__ = 'ieml.ieml_database.descriptors'
```

```
__weakref__
```

list of weak references to the object (if defined)

```
static from_csv_string(s, assert_unique_ieml=False)
```

```
get_descriptor(ieml) → Dict[ieml.constants.DescriptorsType, Dict[ieml.constants.Languages,
List[str]]]
```

```
get_values(ieml, language, descriptor)
```

```
get_values_partial(ieml, language=None, descriptor=None)
```

```
ieml.ieml_database.descriptors.normalize_key(ieml, key, value, parse_ieml=False, par-
tial=False, structure=False)
```

```
class ieml.ieml_database.git_interface.GitInterface(origin='https://github.com/IEMLdev/ieml-
language.git', credentials=<pygit2.credentials.Username
object>, branch='master', com-
mit_id=None, folder=None)
```

Bases: `object`

```
__dict__ = mappingproxy({'__module__': 'ieml.ieml_database.git_interface', '__init__':
```

```
__init__(origin='https://github.com/IEMLdev/ieml-language.git', credentials=<pygit2.credentials.Username
object>, branch='master', commit_id=None,
folder=None)
```

Parameters

- **origin** –
- **credentials** –
- **branch** – the branch to checkout
- **commit_id** – the commit to checkout
- **folder** –

```

__module__ = 'ieml.ieml_database.git_interface'
__weakref__
    list of weak references to the object (if defined)
add_remote (name, url)
checkout (branch=None, commit_id=None)
commit (signature, message)
current_commit
diff (commit0, commit1)
get_version ()
pull (**kwargs)
push (**kwargs)
repo
reset (commit_id=None)
    Set the current branch HEAD to ref the given commit :param commit_id: if set, reset to this commit id,
    otherwise to the head of the branch :return: None
status ()
    ignore path starting with ‘.’
exception ieml.ieml_database.git_interface.MergeConflict (message, conflicts)
    Bases: Exception
__init__ (message, conflicts)
    Initialize self. See help(type(self)) for accurate signature.
__module__ = 'ieml.ieml_database.git_interface'
__repr__ ()
    Return repr(self).
__weakref__
    list of weak references to the object (if defined)
ieml.ieml_database.git_interface.get_local_cache_dir (origin)
class ieml.ieml_database.git_interface.git_transaction (db, signature, message)
    Bases: object
__dict__ = mappingproxy({'__module__': 'ieml.ieml_database.git_interface', '__init__'
__enter__ ()
__exit__ (**kwargs)

```

```

__init__(db, signature, message)
    Initialize self. See help(type(self)) for accurate signature.

__module__ = 'ieml.ieml_database.git_interface'

__weakref__
    list of weak references to the object (if defined)

class ieml.ieml_database.ieml_database.IEMLDatabase(folder, cache_folder=None,
                                                    use_cache=True)
    Bases: object
    CLASS_TO_FOLDER = {<class 'ieml.dictionary.script.script.NullScript'>: ('morpheme', 0
    HASH_SIZE = 10
    MAX_IEML_NAME_SIZE = 100
    __dict__ = mappingproxy({'__module__': 'ieml.ieml_database.ieml_database', 'CLASS_TO_
    __init__(folder, cache_folder=None, use_cache=True)
        Initialize self. See help(type(self)) for accurate signature.
    __module__ = 'ieml.ieml_database.ieml_database'
    __str__()
        Return str(self).
    __weakref__
        list of weak references to the object (if defined)
    add_descriptor(ieml, language, descriptor, value)
    add_structure(ieml, key, value)
    escape_value(v)
    filename_of(ieml)
    get_descriptors(**kwargs)
    get_dictionary(**kwargs)
    get_list(**kwargs)
    get_structure(**kwargs)
    list(**kwargs)
    path_of(_ieml, descriptor=True, mkdir=False, normalize=True)
    remove_descriptor(**kwargs)
    remove_structure(ieml, key=None, value=None, normalize=True)

class ieml.ieml_database.ieml_database.Structure(df)
    Bases: object
    __dict__ = mappingproxy({'__module__': 'ieml.ieml_database.ieml_database', '__init__'
    __init__(df)
        Initialize self. See help(type(self)) for accurate signature.
    __module__ = 'ieml.ieml_database.ieml_database'
    __weakref__
        list of weak references to the object (if defined)
    get_values(**kwargs)

```



```

    get_values_partial (**kwargs)

class ieml.ieml_database.transactions.DBTransaction.DBTransactions (gitdb,
                                                                    signature,
                                                                    cache_folder=None,
                                                                    use_cache=True)

Bases: object

__dict__ = mappingproxy({'__module__': 'ieml.ieml_database.transactions.DBTransaction
__init__ (gitdb, signature, cache_folder=None, use_cache=True)
    Initialize self. See help(type(self)) for accurate signature.

__module__ = 'ieml.ieml_database.transactions.DBTransaction'

__weakref__
    list of weak references to the object (if defined)

add_morpheme_paradigm (script: ieml.dictionary.script.script.Script, translations, comments)

create_root_paradigm (root, inhibitions, translations, comments)

delete_morpheme_paradigm (script: ieml.dictionary.script.script.Script)

delete_morpheme_root_paradigm (script: ieml.dictionary.script.script.Script,
                                empty_descriptors=True)

set_descriptors (ieml, descriptor, value)

set_inhibitions (ieml, inhibitions)

update_all_ieml (f, message: str)

update_morpheme_paradigm (script_old: ieml.dictionary.script.script.Script, script_new:
                            ieml.dictionary.script.script.Script)

ieml.ieml_database.transactions.DBTransaction.append_idx_to_dict (d, idx)

ieml.error (m)

```


i

- [ieuml](#), [29](#)
- [ieuml.common](#)s, [23](#)
- [ieuml.constants](#), [25](#)
- [ieuml.dictionary](#), [9](#)
- [ieuml.dictionary.dictionary](#), [9](#)
- [ieuml.dictionary.script](#), [7](#)
- [ieuml.dictionary.script.operator](#), [8](#)
- [ieuml.dictionary.script.parser](#), [9](#)
- [ieuml.dictionary.script.parser.lexer](#), [9](#)
- [ieuml.dictionary.script.parser.parser](#), [9](#)
- [ieuml.dictionary.script.script](#), [7](#)
- [ieuml.dictionary.script.tools](#), [8](#)
- [ieuml.ieuml_database](#), [26](#)
- [ieuml.ieuml_database.descriptors](#), [26](#)
- [ieuml.ieuml_database.git_interface](#), [26](#)
- [ieuml.ieuml_database.ieuml_database](#), [28](#)
- [ieuml.ieuml_database.transactions.DBTransaction](#),
[29](#)
- [ieuml.usl](#), [11](#)
- [ieuml.usl.constants](#), [23](#)
- [ieuml.usl.decoration](#), [22](#)
- [ieuml.usl.decoration.instance](#), [22](#)
- [ieuml.usl.decoration.parser](#), [21](#)
- [ieuml.usl.decoration.parser.lexer](#), [21](#)
- [ieuml.usl.decoration.parser.parser](#), [21](#)
- [ieuml.usl.decoration.path](#), [18](#)
- [ieuml.usl.lexeme](#), [12](#)
- [ieuml.usl.parser](#), [17](#)
- [ieuml.usl.parser.lexer](#), [17](#)
- [ieuml.usl.parser.parser](#), [17](#)
- [ieuml.usl.polymorpheme](#), [12](#)
- [ieuml.usl.syntagmatic_function](#), [13](#)
- [ieuml.usl.table](#), [16](#)
- [ieuml.usl.usl](#), [11](#)
- [ieuml.usl.variation](#), [16](#)
- [ieuml.usl.word](#), [15](#)

Symbols

- `__add__()` (*ieml.dictionary.script.script.Script* method), 8
- `__bool__()` (*ieml.usl.usl.USL* method), 11
- `__call__()` (*ieml.common.singleton.Singleton* method), 25
- `__call__()` (*ieml.usl.parser.parser.IEMLParserSingleton* method), 18
- `__contains__()` (*ieml.dictionary.dictionary.Dictionary* method), 9
- `__contains__()` (*ieml.dictionary.script.script.Script* method), 8
- `__contains__()` (*ieml.usl.usl.USL* method), 11
- `__dict__` (*ieml.common.decoratedcomponent.DecoratedComponent* attribute), 23
- `__dict__` (*ieml.common.folderwatchercache.FolderWatcherCache* attribute), 24
- `__dict__` (*ieml.common.treestructure.TreeStructure* attribute), 25
- `__dict__` (*ieml.common.cached_property.CachedProperty* attribute), 25
- `__dict__` (*ieml.dictionary.dictionary.Dictionary* attribute), 9
- `__dict__` (*ieml.dictionary.script.parser.parser.ScriptParser* attribute), 9
- `__dict__` (*ieml.ieml_database.descriptors.Descriptors* attribute), 26
- `__dict__` (*ieml.ieml_database.git_interface.GitInterface* attribute), 26
- `__dict__` (*ieml.ieml_database.git_interface.git_transaction.GitTransaction* attribute), 27
- `__dict__` (*ieml.ieml_database.ieml_database.IEMLDatabase* attribute), 28
- `__dict__` (*ieml.ieml_database.ieml_database.Structure* attribute), 28
- `__dict__` (*ieml.ieml_database.transactions.DBTransaction.DBTransaction* attribute), 29
- `__dict__` (*ieml.usl.decoration.instance.Decoration* attribute), 22
- `__dict__` (*ieml.usl.decoration.instance.LiteralContext* attribute), 22
- `__dict__` (*ieml.usl.decoration.parser.parser.PathParser* attribute), 21
- `__dict__` (*ieml.usl.decoration.path.UslPath* attribute), 20
- `__dict__` (*ieml.usl.parser.parser.IEMLParser* attribute), 17
- `__dict__` (*ieml.usl.syntagmatic_function.SyntagmaticFunction* attribute), 14
- `__dict__` (*ieml.usl.syntagmatic_function.SyntagmaticRole* attribute), 15
- `__dict__` (*ieml.usl.table.UslTable2D* attribute), 16
- `__enter__()` (*ieml.ieml_database.git_interface.git_transaction.GitTransaction* method), 27
- `__enter__()` (*ieml.usl.decoration.instance.LiteralContext* method), 22
- `__eq__()` (*ieml.common.treestructure.TreeStructure* method), 25
- `__eq__()` (*ieml.dictionary.script.script.Script* method), 8
- `__eq__()` (*ieml.usl.decoration.instance.Decoration* method), 22
- `__eq__()` (*ieml.usl.decoration.path.UslPath* method), 20
- `__eq__()` (*ieml.usl.syntagmatic_function.SyntagmaticFunction* method), 14
- `__eq__()` (*ieml.usl.syntagmatic_function.SyntagmaticRole* method), 15
- `__eq__()` (*ieml.usl.usl.USL* method), 11
- `__exit__()` (*ieml.ieml_database.git_interface.git_transaction.GitTransaction* method), 27
- `__exit__()` (*ieml.usl.decoration.instance.LiteralContext* method), 22
- `__ge__()` (*ieml.common.orderedenum.OrderedEnum* method), 24
- `__get__()` (*ieml.common.cached_property.CachedProperty* method), 25
- `__getitem__()` (*ieml.dictionary.dictionary.Dictionary* method), 9
- `__getnewargs_ex__()` (*ieml.dictionary.script.script.Script* method), 8
- `__gt__()` (*ieml.common.orderedenum.OrderedEnum* method), 24
- `__hash__` (*ieml.usl.decoration.instance.Decoration* attribute), 22

tribute), 22

__hash__ (iempl.usl.syntagmatic_function.SyntagmaticFunction attribute), 14

__hash__ () (iempl.commons.TreeStructure method), 25

__hash__ () (iempl.dictionary.script.script.Script method), 8

__hash__ () (iempl.usl.decoration.path.UslPath method), 20

__hash__ () (iempl.usl.syntagmatic_function.SyntagmaticRole method), 15

__hash__ () (iempl.usl.usl.USL method), 11

__init__ () (iempl.commons.DecoratedComponent method), 23

__init__ () (iempl.commons.FolderWatcherCache method), 24

__init__ () (iempl.commons.TreeStructure method), 25

__init__ () (iempl.commons.cached_property method), 25

__init__ () (iempl.dictionary.dictionary.Dictionary method), 9

__init__ () (iempl.dictionary.script.parser.parser.ScriptParser method), 9

__init__ () (iempl.dictionary.script.script.AdditiveScript method), 7

__init__ () (iempl.dictionary.script.script.MultiplicativeScript method), 7

__init__ () (iempl.dictionary.script.script.NullScript method), 7

__init__ () (iempl.dictionary.script.script.Script method), 8

__init__ () (iempl.ieml_database.descriptors.Descriptors method), 26

__init__ () (iempl.ieml_database.git_interface.GitInterface method), 26

__init__ () (iempl.ieml_database.git_interface.MergeConflict method), 27

__init__ () (iempl.ieml_database.git_interface.git_transaction method), 27

__init__ () (iempl.ieml_database.ieml_database.IEMLDatabase method), 28

__init__ () (iempl.ieml_database.ieml_database.Structure method), 28

__init__ () (iempl.ieml_database.transactions.DBTransaction method), 29

__init__ () (iempl.usl.decoration.instance.Decoration method), 22

__init__ () (iempl.usl.decoration.instance.InstanceUSL method), 22

__init__ () (iempl.usl.decoration.instance.LiteralContext method), 22

__init__ () (iempl.usl.decoration.parser.parser.PathParser method), 21

__init__ () (iempl.usl.decoration.path.FlexionPath method), 18

__init__ () (iempl.usl.decoration.path.LexemePath method), 19

__init__ () (iempl.usl.decoration.path.PolymorphemePath method), 19

__init__ () (iempl.usl.decoration.path.RolePath method), 20

__init__ () (iempl.usl.decoration.path.UslPath method), 20

__init__ () (iempl.usl.lexeme.Lexeme method), 13

__init__ () (iempl.usl.parser.parser.IEMLParser method), 17

__init__ () (iempl.usl.polymorpheme.PolyMorpheme method), 12

__init__ () (iempl.usl.syntagmatic_function.DependantQualitySyntagmatic method), 13

__init__ () (iempl.usl.syntagmatic_function.IndependantQualitySyntagmatic method), 13

__init__ () (iempl.usl.syntagmatic_function.JunctionSyntagmaticFunction method), 14

__init__ () (iempl.usl.syntagmatic_function.ProcessSyntagmaticFunction method), 14

__init__ () (iempl.usl.syntagmatic_function.SyntagmaticFunction method), 14

__init__ () (iempl.usl.syntagmatic_function.SyntagmaticRole method), 15

__init__ () (iempl.usl.table.UslTable2D method), 16

__init__ () (iempl.usl.usl.USL method), 11

__init__ () (iempl.usl.variation.PolyMorphemeVariation method), 16

__init__ () (iempl.usl.word.Word method), 15

__iter__ () (iempl.commons.TreeStructure method), 25

__iter__ () (iempl.dictionary.script.script.NullScript method), 7

__le__ () (iempl.commons.OrderedEnum method), 24

__len__ () (iempl.dictionary.dictionary.Dictionary method), 9

__len__ () (iempl.dictionary.script.script.Script method), 8

__lt__ () (iempl.usl.usl.USL method), 11

__lt__ () (iempl.commons.OrderedEnum method), 24

__lt__ () (iempl.dictionary.script.script.Script method), 8

__lt__ () (iempl.usl.decoration.instance.Decoration method), 22

__lt__ () (iempl.usl.decoration.path.UslPath method), 20

__lt__ () (iempl.usl.syntagmatic_function.SyntagmaticFunction method), 14

__lt__ () (iempl.usl.syntagmatic_function.SyntagmaticRole method), 15

__lt__ () (iempl.usl.usl.USL method), 11

__module__ (iempl.commons.DecoratedComponent attribute), 23

__module__ (iempl.commons.FolderWatcherCache at-

[tribute](#)), 24
[__module__](#) ([iempl.commons.LastUpdatedOrderedDict](#) [attribute](#)), 24
[__module__](#) ([iempl.commons.OrderedEnum](#) [attribute](#)), 25
[__module__](#) ([iempl.commons.Singleton](#) [attribute](#)), 25
[__module__](#) ([iempl.commons.TreeStructure](#) [attribute](#)), 25
[__module__](#) ([iempl.commons.cached_property](#) [attribute](#)), 25
[__module__](#) ([iempl.constants.DescriptorsType](#) [attribute](#)), 26
[__module__](#) ([iempl.constants.Languages](#) [attribute](#)), 26
[__module__](#) ([iempl.dictionary.dictionary.Dictionary](#) [attribute](#)), 9
[__module__](#) ([iempl.dictionary.script.parser.parser.ScriptParser](#) [attribute](#)), 9
[__module__](#) ([iempl.dictionary.script.script.AdditiveScript](#) [attribute](#)), 7
[__module__](#) ([iempl.dictionary.script.script.MultiplicativeScript](#) [attribute](#)), 7
[__module__](#) ([iempl.dictionary.script.script.NullScript](#) [attribute](#)), 7
[__module__](#) ([iempl.dictionary.script.script.Script](#) [attribute](#)), 8
[__module__](#) ([iempl.ieml_database.descriptors.Descriptors](#) [attribute](#)), 26
[__module__](#) ([iempl.ieml_database.git_interface.GitInterface](#) [attribute](#)), 27
[__module__](#) ([iempl.ieml_database.git_interface.MergeConflict](#) [attribute](#)), 27
[__module__](#) ([iempl.ieml_database.git_interface.git_transaction](#) [attribute](#)), 28
[__module__](#) ([iempl.ieml_database.ieml_database.IEMLDatabase](#) [attribute](#)), 28
[__module__](#) ([iempl.ieml_database.ieml_database.Structure](#) [attribute](#)), 28
[__module__](#) ([iempl.ieml_database.transactions.DBTransaction](#) [attribute](#)), 29
[__module__](#) ([iempl.usl.decoration.instance.Decoration](#) [attribute](#)), 22
[__module__](#) ([iempl.usl.decoration.instance.InstanceUSL](#) [attribute](#)), 22
[__module__](#) ([iempl.usl.decoration.instance.LiteralContext](#) [attribute](#)), 23
[__module__](#) ([iempl.usl.decoration.parser.parser.PathParser](#) [attribute](#)), 21
[__module__](#) ([iempl.usl.decoration.path.DeferenceError](#) [attribute](#)), 18
[__module__](#) ([iempl.usl.decoration.path.FlexionPath](#) [attribute](#)), 18
[__module__](#) ([iempl.usl.decoration.path.GroupIndex](#) [attribute](#)), 19
[__module__](#) ([iempl.usl.decoration.path.LexemeIndex](#) [attribute](#)), 19
[__module__](#) ([iempl.usl.decoration.path.LexemePath](#) [attribute](#)), 19
[__module__](#) ([iempl.usl.decoration.path.PolymorphemePath](#) [attribute](#)), 19
[__module__](#) ([iempl.usl.decoration.path.RolePath](#) [attribute](#)), 20
[__module__](#) ([iempl.usl.decoration.path.UslPath](#) [attribute](#)), 20
[__module__](#) ([iempl.usl.lexeme.Lexeme](#) [attribute](#)), 13
[__module__](#) ([iempl.usl.parser.parser.IEMLParser](#) [attribute](#)), 17
[__module__](#) ([iempl.usl.parser.parser.IEMLParserSingleton](#) [attribute](#)), 18
[__module__](#) ([iempl.usl.polymorpheme.PolyMorpheme](#) [attribute](#)), 12
[__module__](#) ([iempl.usl.syntagmatic_function.DependantQualitySyntagmatic](#) [attribute](#)), 13
[__module__](#) ([iempl.usl.syntagmatic_function.IndependantQualitySyntagmatic](#) [attribute](#)), 14
[__module__](#) ([iempl.usl.syntagmatic_function.JunctionSyntagmaticFunction](#) [attribute](#)), 14
[__module__](#) ([iempl.usl.syntagmatic_function.ProcessSyntagmaticFunction](#) [attribute](#)), 14
[__module__](#) ([iempl.usl.syntagmatic_function.SyntagmaticFunction](#) [attribute](#)), 14
[__module__](#) ([iempl.usl.syntagmatic_function.SyntagmaticRole](#) [attribute](#)), 15
[__module__](#) ([iempl.usl.table.UslTable2D](#) [attribute](#)), 16
[__module__](#) ([iempl.usl.usl.USL](#) [attribute](#)), 11
[__module__](#) ([iempl.usl.variation.PolyMorphemeVariation](#) [attribute](#)), 16
[__module__](#) ([iempl.usl.variation.Variation](#) [attribute](#)), 17
[__module__](#) ([iempl.usl.word.Word](#) [attribute](#)), 15
[__new__](#) () ([iempl.commons.TreeStructure](#) [method](#)), 25
[__new__](#) () ([iempl.dictionary.script.script.Script](#) [static method](#)), 8
[__new__](#) () ([iempl.ieml_database.git_interface.MergeConflict](#) [method](#)), 27
[__setitem__](#) () ([iempl.commons.LastUpdatedOrderedDict](#) [method](#)), 24
[__str__](#) () ([iempl.commons.TreeStructure](#) [method](#)), 25
[__str__](#) () ([iempl.ieml_database.ieml_database.IEMLDatabase](#) [method](#)), 28
[__str__](#) () ([iempl.usl.decoration.instance.Decoration](#) [method](#)), 22
[__str__](#) () ([iempl.usl.decoration.instance.InstanceUSL](#) [method](#)), 22
[__str__](#) () ([iempl.usl.decoration.path.UslPath](#) [method](#)), 20
[__str__](#) () ([iempl.usl.syntagmatic_function.SyntagmaticRole](#) [method](#)), 15
[__str__](#) () ([iempl.usl.usl.USL](#) [method](#)), 11
[__weakref__](#) ([iempl.commons.DecoratedComponent](#)

[attribute](#)), 23
[__weakref__](#) ([ieml.common.FolderWatcherCache](#) attribute), 24
[__weakref__](#) ([ieml.common.TreeStructure](#) attribute), 25
[__weakref__](#) ([ieml.common.cached_property](#) attribute), 25
[__weakref__](#) ([ieml.dictionary.dictionary.Dictionary](#) attribute), 9
[__weakref__](#) ([ieml.dictionary.script.parser.parser.ScriptParser](#) attribute), 9
[__weakref__](#) ([ieml.ieml_database.descriptors.Descriptors](#) attribute), 26
[__weakref__](#) ([ieml.ieml_database.git_interface.GitInterface](#) attribute), 27
[__weakref__](#) ([ieml.ieml_database.git_interface.MergeConflict](#) attribute), 27
[__weakref__](#) ([ieml.ieml_database.git_interface.git_transaction](#) attribute), 28
[__weakref__](#) ([ieml.ieml_database.ieml_database.IEMLDatabase](#) attribute), 28
[__weakref__](#) ([ieml.ieml_database.ieml_database.Structure](#) attribute), 28
[__weakref__](#) ([ieml.ieml_database.transactions.DBTransaction](#) attribute), 29
[__weakref__](#) ([ieml.usl.decoration.instance.Decoration](#) attribute), 22
[__weakref__](#) ([ieml.usl.decoration.instance.LiteralContext](#) attribute), 23
[__weakref__](#) ([ieml.usl.decoration.parser.parser.PathParser](#) attribute), 21
[__weakref__](#) ([ieml.usl.decoration.path.DeferenceError](#) attribute), 18
[__weakref__](#) ([ieml.usl.decoration.path.UslPath](#) attribute), 20
[__weakref__](#) ([ieml.usl.parser.parser.IEMLParser](#) attribute), 17
[__weakref__](#) ([ieml.usl.syntagmatic_function.SyntagmaticFunction](#) attribute), 14
[__weakref__](#) ([ieml.usl.syntagmatic_function.SyntagmaticRole](#) attribute), 15
[__weakref__](#) ([ieml.usl.table.UslTable2D](#) attribute), 16

A

[add\(\)](#) (in module [ieml.dictionary.script.operator](#)), 8
[add_descriptor\(\)](#) ([ieml.ieml_database.ieml_database.IEMLDatabase](#) method), 28
[add_morpheme_paradigm\(\)](#) ([ieml.ieml_database.transactions.DBTransaction.DBTransactions](#) method), 29
[add_remote\(\)](#) ([ieml.ieml_database.git_interface.GitInterface](#) method), 27
[add_structure\(\)](#) ([ieml.ieml_database.ieml_database.IEMLDatabase](#) method), 28

[AdditiveScript](#) (class in [ieml.dictionary.script.script](#)), 7
[append_idx_to_dict\(\)](#) (in module [ieml.ieml_database.transactions.DBTransaction](#)), 29
[apply\(\)](#) ([ieml.usl.decoration.instance.Decoration](#) method), 22
[as_constant\(\)](#) ([ieml.usl.decoration.path.FlexionPath](#) method), 18
[as_constant\(\)](#) ([ieml.usl.decoration.path.LexemePath](#) method), 19
[as_constant\(\)](#) ([ieml.usl.decoration.path.PolymorphemePath](#) method), 19
[as_constant\(\)](#) ([ieml.usl.decoration.path.RolePath](#) method), 20
[as_constant\(\)](#) ([ieml.usl.decoration.path.UslPath](#) method), 20
[assert_in\(\)](#) ([ieml.usl.syntagmatic_function.SyntagmaticFunction](#) method), 14
[assert_in\(\)](#) (in module [ieml.usl.constants](#)), 23
[assert_all_in\(\)](#) (in module [ieml.usl.constants](#)), 23
[assert_atmost_one_from\(\)](#) (in module [ieml.usl.constants](#)), 23
[assert_atmost_one_from\(\)](#) (in module [ieml.usl.constants](#)), 23
[assert_only_one_from\(\)](#) (in module [ieml.usl.constants](#)), 23

B

[build_usl_from_path_to_node\(\)](#) ([ieml.usl.decoration.path.FlexionPath](#) class method), 18
[build_usl_from_path_to_node\(\)](#) ([ieml.usl.decoration.path.LexemePath](#) class method), 19
[build_usl_from_path_to_node\(\)](#) ([ieml.usl.decoration.path.PolymorphemePath](#) class method), 19
[build_usl_from_path_to_node\(\)](#) ([ieml.usl.decoration.path.RolePath](#) class method), 20
[build_usl_from_path_to_node\(\)](#) ([ieml.usl.decoration.path.UslPath](#) class method), 20

C

[cache_file](#) ([ieml.common.FolderWatcherCache](#) attribute), 24
[cache_results_watch_files\(\)](#) (in module [ieml.common](#)), 25
[cached_property](#) (class in [ieml.common](#)), 25
[cardinal](#) ([ieml.usl.usl.USL](#) attribute), 11
[cells](#) ([ieml.usl.table.UslTable2D](#) attribute), 16

[check \(\) \(ieml.dictionary.script.script.Script method\), 8](#)
[check \(\) \(ieml.usl.decoration.instance.InstancedUSL method\), 22](#)
[check \(\) \(ieml.usl.lexeme.Lexeme method\), 13](#)
[check \(\) \(ieml.usl.polymorpheme.PolyMorpheme method\), 12](#)
[check \(\) \(ieml.usl.syntagmatic_function.DependantQualitySyntagmaticFunction method\), 13](#)
[check \(\) \(ieml.usl.syntagmatic_function.JunctionSyntagmaticFunction method\), 14](#)
[check \(\) \(ieml.usl.syntagmatic_function.ProcessSyntagmaticFunction method\), 14](#)
[check \(\) \(ieml.usl.syntagmatic_function.SyntagmaticFunction method\), 14](#)
[check \(\) \(ieml.usl.usl.USL method\), 11](#)
[check \(\) \(ieml.usl.variation.PolyMorphemeVariation method\), 16](#)
[check \(\) \(ieml.usl.word.Word method\), 16](#)
[check_address_script \(\) \(in module ieml.usl.constants\), 23](#)
[check_flexion_actant_scripts \(\) \(in module ieml.usl.constants\), 23](#)
[check_flexion_process_scripts \(\) \(in module ieml.usl.constants\), 23](#)
[check_flexion_quality \(\) \(in module ieml.usl.constants\), 23](#)
[check_lexeme \(\) \(in module ieml.usl.lexeme\), 13](#)
[check_lexeme_scripts \(\) \(in module ieml.usl.constants\), 23](#)
[check_polymorpheme \(\) \(in module ieml.usl.polymorpheme\), 12](#)
[check_word \(\) \(in module ieml.usl.word\), 16](#)
[checkout \(\) \(ieml.ieml_database.git_interface.GitInterface method\), 27](#)
[class_from_address \(\) \(in module ieml.usl.constants\), 23](#)
[CLASS_TO_FOLDER \(ieml.ieml_database.ieml_database.IEMLDatabase attribute\), 28](#)
[clear_literal \(\) \(ieml.common.DecoratedComponent method\), 23](#)
[clone \(\) \(ieml.usl.decoration.path.FlexionPath method\), 18](#)
[clone \(\) \(ieml.usl.decoration.path.LexemePath method\), 19](#)
[clone \(\) \(ieml.usl.decoration.path.PolymorphemePath method\), 19](#)
[clone \(\) \(ieml.usl.decoration.path.RolePath method\), 20](#)
[clone \(\) \(ieml.usl.decoration.path.UsPath method\), 20](#)
[column_paths_constant \(ieml.usl.table.UsTable2D attribute\), 16](#)
[column_paths_variation \(ieml.usl.table.UsTable2D attribute\), 16](#)
[columns \(ieml.usl.table.UsTable2D attribute\), 16](#)
[COMMENTS \(ieml.constants.DescriptorsType attribute\), 26](#)
[commit \(\) \(ieml.ieml_database.git_interface.GitInterface method\), 27](#)
[compute_PM_singular_sequences \(\) \(in module ieml.usl.polymorpheme\), 12](#)
[create_usl_decoration_path \(ieml.usl.decoration.path.UsPath method\), 20](#)
[create_usl_group_index \(ieml.usl.decoration.path.GroupIndex attribute\), 19](#)
[create_usl_paths \(ieml.usl.table.UsTable2D attribute\), 16](#)
[contained \(\) \(ieml.usl.decoration.path.UsPath method\), 20](#)
[CONTENT \(ieml.usl.decoration.path.LexemeIndex attribute\), 19](#)
[create_root_paradigm \(\) \(ieml.ieml_database.transactions.DBTransaction.DBTransactions method\), 29](#)
[current_commit \(ieml.ieml_database.git_interface.GitInterface attribute\), 27](#)

D

[DBTransactions \(class in ieml.ieml_database.transactions.DBTransaction\), 29](#)
[DecoratedComponent \(class in ieml.common\), 23](#)
[Decoration \(class in ieml.usl.decoration.instance\), 22](#)
[deference \(\) \(ieml.usl.decoration.path.UsPath method\), 20](#)
[DeferenceError, 18](#)
[delete_morpheme_paradigm \(\) \(ieml.ieml_database.transactions.DBTransaction.DBTransactions method\), 29](#)
[delete_morpheme_root_paradigm \(\) \(ieml.ieml_database.transactions.DBTransaction.DBTransactions method\), 29](#)
[DependantQualitySyntagmaticFunction \(class in ieml.usl.syntagmatic_function\), 13](#)
[Descriptors \(class in ieml.ieml_database.descriptors\), 26](#)
[DescriptorsType \(class in ieml.constants\), 25](#)
[Dictionary \(class in ieml.dictionary.dictionary\), 9](#)
[diff \(\) \(ieml.ieml_database.git_interface.GitInterface method\), 27](#)
[do_lt \(\) \(ieml.usl.decoration.instance.InstancedUSL method\), 22](#)
[do_lt \(\) \(ieml.usl.lexeme.Lexeme method\), 13](#)
[do_lt \(\) \(ieml.usl.polymorpheme.PolyMorpheme method\), 12](#)
[do_lt \(\) \(ieml.usl.usl.USL method\), 11](#)
[do_lt \(\) \(ieml.usl.variation.PolyMorphemeVariation method\), 16](#)
[do_lt \(\) \(ieml.usl.word.Word method\), 16](#)

E

`empty` (*ieml.usl.decoration.instance.InstancedUSL attribute*), 22

`empty` (*ieml.usl.lexeme.Lexeme attribute*), 13

`empty` (*ieml.usl.polymorpheme.PolyMorpheme attribute*), 12

`empty` (*ieml.usl.syntagmatic_function.SyntagmaticFunction attribute*), 14

`empty` (*ieml.usl.usl.USL attribute*), 11

`empty` (*ieml.usl.variation.PolyMorphemeVariation attribute*), 17

`empty` (*ieml.usl.word.Word attribute*), 16

`EN` (*ieml.constants.Languages attribute*), 26

`enumerate_partitions()` (*in module ieml.usl.table*), 16

`error()` (*in module ieml*), 29

`escape_value()` (*ieml.ieml_database.ieml_database.IEMLDatabase method*), 28

F

`factor()` (*in module ieml.dictionary.script.tools*), 8

`factorize()` (*in module ieml.dictionary.script.tools*), 8

`filename_of()` (*ieml.ieml_database.ieml_database.IEMLDatabase method*), 28

`FLEXION` (*ieml.usl.decoration.path.LexemeIndex attribute*), 19

`FlexionPath` (*class in ieml.usl.decoration.path*), 18

`FolderWatcherCache` (*class in ieml.common*), 24

`FR` (*ieml.constants.Languages attribute*), 26

`from_csv_string()` (*ieml.ieml_database.descriptors.Descriptors static method*), 26

`from_list()` (*ieml.usl.syntagmatic_function.SyntagmaticFunction static method*), 14

`from_string()` (*ieml.usl.decoration.path.UslPath class method*), 20

`from_usl()` (*ieml.usl.decoration.instance.InstancedUSL static method*), 22

`fullname()` (*in module ieml.common*), 25

G

`get()` (*ieml.common.FolderWatcherCache method*), 24

`get()` (*ieml.usl.syntagmatic_function.SyntagmaticFunction method*), 15

`get_context_role_prefix()` (*ieml.usl.syntagmatic_function.SyntagmaticFunction static method*), 15

`get_descriptor()` (*ieml.ieml_database.descriptors.Descriptors method*), 26

`get_descriptors()` (*ieml.ieml_database.ieml_database.IEMLDatabase method*), 28

`get_dictionary()` (*ieml.ieml_database.ieml_database.IEMLDatabase method*), 28

`get_iemldb_folder()` (*in module ieml.constants*), 26

`get_index()` (*in module ieml.usl*), 11

`get_lexer()` (*in module ieml.usl.decoration.parser.lexer*), 21

`get_lexer()` (*in module ieml.usl.parser.lexer*), 17

`get_list()` (*ieml.ieml_database.ieml_database.IEMLDatabase method*), 28

`get_literal()` (*ieml.common.DecoratedComponent method*), 23

`get_local_cache_dir()` (*in module ieml.ieml_database.git_interface*), 27

`get_paradigm()` (*ieml.usl.syntagmatic_function.SyntagmaticFunction method*), 15

`get_role_expansion()` (*ieml.usl.syntagmatic_function.SyntagmaticFunction method*), 15

`get_script_lexer()` (*in module ieml.dictionary.script.parser.lexer*), 9

`get_structure()` (*ieml.ieml_database.ieml_database.IEMLDatabase method*), 28

`get_values()` (*ieml.ieml_database.descriptors.Descriptors method*), 26

`get_values()` (*ieml.ieml_database.ieml_database.Structure method*), 28

`get_values_partial()` (*ieml.ieml_database.descriptors.Descriptors method*), 26

`get_values_partial()` (*ieml.ieml_database.ieml_database.Structure method*), 29

`get_version()` (*ieml.ieml_database.git_interface.GitInterface method*), 27

`git_transaction` (*class in ieml.ieml_database.git_interface*), 27

`GitInterface` (*class in ieml.ieml_database.git_interface*), 26

`GROUP_0` (*ieml.usl.decoration.path.GroupIndex attribute*), 19

`GROUP_1` (*ieml.usl.decoration.path.GroupIndex attribute*), 19

`GROUP_2` (*ieml.usl.decoration.path.GroupIndex attribute*), 19

`GroupIndex` (*class in ieml.usl.decoration.path*), 18

`has_prefix()` (*ieml.usl.decoration.path.UslPath method*), 20

`HASH_SIZE` (*ieml.ieml_database.ieml_database.IEMLDatabase attribute*), 28

`headers` (*ieml.dictionary.script.script.Script attribute*), 8

I

ieml (module), 29
*ieml.common*s (module), 23
ieml.constants (module), 25
ieml.dictionary (module), 9
ieml.dictionary.dictionary (module), 9
ieml.dictionary.script (module), 7
ieml.dictionary.script.operator (module), 8
ieml.dictionary.script.parser (module), 9
ieml.dictionary.script.parser.lexer (module), 9
ieml.dictionary.script.parser.parser (module), 9
ieml.dictionary.script.script (module), 7
ieml.dictionary.script.tools (module), 8
ieml.ieml_database (module), 26
ieml.ieml_database.descriptors (module), 26
ieml.ieml_database.git_interface (module), 26
ieml.ieml_database.ieml_database (module), 28
ieml.ieml_database.transactions.DBTransaction (module), 29
ieml.usl (module), 11
ieml.usl.constants (module), 23
ieml.usl.decoration (module), 22
ieml.usl.decoration.instance (module), 22
ieml.usl.decoration.parser (module), 21
ieml.usl.decoration.parser.lexer (module), 21
ieml.usl.decoration.parser.parser (module), 21
ieml.usl.decoration.path (module), 18
ieml.usl.lexeme (module), 12
ieml.usl.parser (module), 17
ieml.usl.parser.lexer (module), 17
ieml.usl.parser.parser (module), 17
ieml.usl.polymorpheme (module), 12
ieml.usl.syntagmatic_function (module), 13
ieml.usl.table (module), 16
ieml.usl.usl (module), 11
ieml.usl.variation (module), 16
ieml.usl.word (module), 15
IEMLDatabase (class in *ieml.ieml_database.ieml_database*), 28
IEMLParser (class in *ieml.usl.parser.parser*), 17
IEMLParserSingleton (class in *ieml.usl.parser.parser*), 18
IndependantQualitySyntagmaticFunction (class in *ieml.usl.syntagmatic_function*), 13
InstancedUSL (class in *ieml.usl.decoration.instance*), 22
int2base() (in module *ieml.usl*), 11
is_constant_path(*ieml.usl.decoration.path.USLPath* attribute), 20
is_junction_prefix() (*ieml.usl.syntagmatic_function.SyntagmaticRole* method), 15
is_pruned() (*ieml.common*s.FolderWatcherCache method), 24
is_singular(*ieml.dictionary.script.script.Script* attribute), 8
is_singular(*ieml.usl.usl.USL* attribute), 11
iter_structure() (*ieml.dictionary.script.script.Script* method), 8
iter_structure() (*ieml.usl.decoration.instance.InstancedUSL* method), 22
iter_structure() (*ieml.usl.lexeme.Lexeme* method), 13
iter_structure() (*ieml.usl.polymorpheme.PolyMorpheme* method), 12
iter_structure() (*ieml.usl.syntagmatic_function.SyntagmaticFunction* method), 15
iter_structure() (*ieml.usl.usl.USL* method), 11
iter_structure() (*ieml.usl.variation.PolyMorphemeVariation* method), 17
iter_structure() (*ieml.usl.word.Word* method), 16
iter_structure_path() (*ieml.dictionary.script.script.Script* method), 8
iter_structure_path() (*ieml.usl.decoration.instance.InstancedUSL* method), 22
iter_structure_path() (*ieml.usl.lexeme.Lexeme* method), 13
iter_structure_path() (*ieml.usl.polymorpheme.PolyMorpheme* method), 12
iter_structure_path() (*ieml.usl.syntagmatic_function.SyntagmaticFunction* method), 15
iter_structure_path() (*ieml.usl.usl.USL* method), 12
iter_structure_path() (*ieml.usl.variation.PolyMorphemeVariation* method), 17
iter_structure_path() (*ieml.usl.word.Word* method), 16
in iter_structure_path_by_script_ss() (*ieml.dictionary.script.script.Script* method), 8
iter_structure_path_by_script_ss() (*ieml.usl.usl.USL* method), 12
iter_structure_path_by_type() (*ieml.usl.usl.USL* method), 12
J
JunctionSyntagmaticFunction (class in

ieml.usl.syntagmatic_function), 14

L

Languages (*class in ieml.constants*), 26

LastUpdatedOrderedDict (*class in ieml.common*s), 24

Lexeme (*class in ieml.usl.lexeme*), 12

LexemeIndex (*class in ieml.usl.decoration.path*), 19

LexemePath (*class in ieml.usl.decoration.path*), 19

list() (*ieml.ieml_database.ieml_database.IEMLDatabase method*), 28

list_decorations() (*ieml.usl.decoration.instance.InstancedUSL static method*), 22

literal_context() (*in module ieml.usl.decoration.instance*), 23

LiteralContext (*class in ieml.usl.decoration.instance*), 22

lock (*ieml.dictionary.script.parser.parser.ScriptParser attribute*), 9

lock (*ieml.usl.decoration.parser.parser.PathParser attribute*), 21

lock (*ieml.usl.parser.parser.IEMLParser attribute*), 17

M

m() (*in module ieml.dictionary.script.operator*), 8

MAX_IEML_NAME_SIZE (*ieml.ieml_database.ieml_database.IEMLDatabase attribute*), 28

MergeConflict, 27

monitor_decorator() (*in module ieml.common*s), 25

morphemes (*ieml.usl.decoration.instance.InstancedUSL attribute*), 22

morphemes (*ieml.usl.lexeme.Lexeme attribute*), 13

morphemes (*ieml.usl.polymorpheme.PolyMorpheme attribute*), 12

morphemes (*ieml.usl.usl.USL attribute*), 12

morphemes (*ieml.usl.variation.PolyMorphemeVariation attribute*), 17

morphemes (*ieml.usl.word.Word attribute*), 16

MultiplicativeScript (*class in ieml.dictionary.script.script*), 7

N

no_child_clone() (*ieml.usl.decoration.path.UslPath method*), 20

normalize_key() (*in module ieml.ieml_database.descriptors*), 26

NullScript (*class in ieml.dictionary.script.script*), 7

O

OrderedEnum (*class in ieml.common*s), 24

P

p_additive_script_lvl_0()

(*ieml.dictionary.script.parser.parser.ScriptParser method*), 9

p_additive_script_lvl_1()

(*ieml.dictionary.script.parser.parser.ScriptParser method*), 9

p_additive_script_lvl_2()

(*ieml.dictionary.script.parser.parser.ScriptParser method*), 9

p_additive_script_lvl_3()

(*ieml.dictionary.script.parser.parser.ScriptParser method*), 10

p_additive_script_lvl_4()

(*ieml.dictionary.script.parser.parser.ScriptParser method*), 10

p_additive_script_lvl_5()

(*ieml.dictionary.script.parser.parser.ScriptParser method*), 10

p_additive_script_lvl_6()

(*ieml.dictionary.script.parser.parser.ScriptParser method*), 10

p_decoration() (*ieml.usl.parser.parser.IEMLParser method*), 17

p_decoration_list()

(*ieml.usl.parser.parser.IEMLParser method*), 17

p_error() (*ieml.dictionary.script.parser.parser.ScriptParser method*), 10

p_error() (*ieml.usl.decoration.parser.parser.PathParser method*), 21

p_error() (*ieml.usl.parser.parser.IEMLParser method*), 17

p_flexion_path() (*ieml.usl.decoration.parser.parser.PathParser method*), 21

p_group() (*ieml.usl.parser.parser.IEMLParser method*), 17

p_group_list() (*ieml.usl.parser.parser.IEMLParser method*), 17

p_ieml_proposition()

(*ieml.usl.parser.parser.IEMLParser method*), 17

p_instanced_usl()

(*ieml.usl.parser.parser.IEMLParser method*), 17

p_lexeme() (*ieml.usl.parser.parser.IEMLParser method*), 17

p_lexeme_list() (*ieml.usl.parser.parser.IEMLParser method*), 17

p_lexeme_path() (*ieml.usl.decoration.parser.parser.PathParser method*), 21

p_morpheme() (*ieml.usl.parser.parser.IEMLParser method*), 18

p_morpheme_sum() (*ieml.usl.parser.parser.IEMLParser*

method), 18
 p_path() (*ieml.usl.decoration.parser.parser.PathParser* *method*), 21
 p_poly_morpheme() (*ieml.usl.parser.parser.IEMLParser* *method*), 18
 p_polymorpheme_path() (*ieml.usl.decoration.parser.parser.PathParser* *method*), 21
 p_positioned_lexeme() (*ieml.usl.parser.parser.IEMLParser* *method*), 18
 p_role_path() (*ieml.usl.decoration.parser.parser.PathParser* *method*), 21
 p_role_path_list() (*ieml.usl.decoration.parser.parser.PathParser* *method*), 21
 p_script_lvl_0() (*ieml.dictionary.script.parser.parser.ScriptParser* *method*), 10
 p_script_lvl_1() (*ieml.dictionary.script.parser.parser.ScriptParser* *method*), 10
 p_script_lvl_2() (*ieml.dictionary.script.parser.parser.ScriptParser* *method*), 10
 p_script_lvl_3() (*ieml.dictionary.script.parser.parser.ScriptParser* *method*), 10
 p_script_lvl_4() (*ieml.dictionary.script.parser.parser.ScriptParser* *method*), 10
 p_script_lvl_5() (*ieml.dictionary.script.parser.parser.ScriptParser* *method*), 10
 p_script_lvl_6() (*ieml.dictionary.script.parser.parser.ScriptParser* *method*), 10
 p_sum_lvl_0() (*ieml.dictionary.script.parser.parser.ScriptParser* *method*), 10
 p_sum_lvl_1() (*ieml.dictionary.script.parser.parser.ScriptParser* *method*), 10
 p_sum_lvl_2() (*ieml.dictionary.script.parser.parser.ScriptParser* *method*), 10
 p_sum_lvl_3() (*ieml.dictionary.script.parser.parser.ScriptParser* *method*), 10
 p_sum_lvl_4() (*ieml.dictionary.script.parser.parser.ScriptParser* *method*), 10
 p_sum_lvl_5() (*ieml.dictionary.script.parser.parser.ScriptParser* *method*), 10
 p_sum_lvl_6() (*ieml.dictionary.script.parser.parser.ScriptParser* *method*), 10
 p_term() (*ieml.dictionary.script.parser.parser.ScriptParser* *method*), 11
 p_usl() (*ieml.usl.parser.parser.IEMLParser* *method*), 18
 p_word() (*ieml.usl.parser.parser.IEMLParser* *method*), 18
 pack_factorisation() (*in module ieml.dictionary.script.tools*), 9
 parse() (*ieml.usl.decoration.parser.parser.PathParser* *method*), 21
 parse() (*ieml.usl.parser.parser.IEMLParser* *method*), 18
 path() (*in module ieml.usl.decoration.path*), 21
 path_of() (*ieml.ieml_database.ieml_database.IEMLDatabase* *method*), 28
 PathParser (*class in ieml.usl.decoration.parser.parser*), 21
 PolyMorpheme (*class in ieml.usl.polymorpheme*), 12
 PolymorphemePath (*class in ieml.usl.decoration.path*), 19
 PolyMorphemeVariation (*class in ieml.usl.variation*), 16
 ProcessSyntagmaticFunction (*class in ieml.usl.syntagmatic_function*), 14
 promote() (*in module ieml.dictionary.script.tools*), 9
 pull() (*ieml.ieml_database.git_interface.GitInterface* *method*), 27
 push() (*ieml.ieml_database.git_interface.GitInterface* *method*), 27
 push() (*ieml.usl.decoration.instance.LiteralContext* *method*), 23

R

REFERENTIAL (*ieml.constants.DescriptorsType* *attribute*), 26
 remove_descriptor() (*ieml.ieml_database.ieml_database.IEMLDatabase* *method*), 28
 remove_prefix() (*ieml.usl.decoration.path.UslPath* *method*), 21
 remove_structure() (*ieml.ieml_database.ieml_database.IEMLDatabase* *method*), 28
 render_with_context() (*ieml.usl.syntagmatic_function.SyntagmaticFunction* *method*), 15
 reset() (*ieml.ieml_database.git_interface.GitInterface* *attribute*), 27
 reset() (*ieml.ieml_database.git_interface.GitInterface* *method*), 27
 reset_in() (*ieml.usl.syntagmatic_function.SyntagmaticFunction* *method*), 15
 reset_junction() (*ieml.usl.syntagmatic_function.SyntagmaticFunction* *method*), 15
 RolePath (*class in ieml.usl.decoration.path*), 20
 row_paths_constant (*ieml.usl.table.UslTable2D* *attribute*), 16
 row_paths_variation (*ieml.usl.table.UslTable2D* *attribute*), 16
 rows (*ieml.usl.table.UslTable2D* *attribute*), 16

S

Script (class in *ieml.dictionary.script.script*), 8
 script() (in module *ieml.dictionary.script.operator*), 8
 ScriptParser (class in *ieml.dictionary.script.parser.parser*), 9
 set_descriptors() (*ieml.ieml_database.transactions.DBTransaction.DBTransactions* method), 29
 set_inhibitions() (*ieml.ieml_database.transactions.DBTransaction.DBTransactions* method), 29
 set_literal() (*ieml.commons.DecoratedComponent* method), 24
 simplify_word() (in module *ieml.usl.word*), 16
 Singleton (class in *ieml.commons*), 25
 singular_sequences (*ieml.dictionary.script.script.Script* attribute), 8
 singular_sequences (*ieml.usl.usl.USL* attribute), 12
 singular_sequences() (*ieml.usl.syntagmatic_function.SyntagmaticFunction* method), 15
 singular_sequences_set (*ieml.dictionary.script.script.Script* attribute), 8
 singular_sequences_set (*ieml.usl.usl.USL* attribute), 12
 split_tail() (*ieml.usl.decoration.path.UslPath* method), 21
 status() (*ieml.ieml_database.git_interface.GitInterface* method), 27
 Structure (class in *ieml.ieml_database.ieml_database*), 28
 syntactic_level (*ieml.usl.decoration.instance.InstanceUSL* attribute), 22
 syntactic_level (*ieml.usl.lexeme.Lexeme* attribute), 13
 syntactic_level (*ieml.usl.polymorpheme.PolyMorpheme* attribute), 12
 syntactic_level (*ieml.usl.usl.USL* attribute), 12
 syntactic_level (*ieml.usl.word.Word* attribute), 16
 SyntagmaticFunction (class in *ieml.usl.syntagmatic_function*), 14
 SyntagmaticRole (class in *ieml.usl.syntagmatic_function*), 15

T

t_add_rules() (*ieml.dictionary.script.parser.parser.ScriptParser* method), 11
 t_parse (*ieml.dictionary.script.parser.parser.ScriptParser* attribute), 11

tables_script (*ieml.dictionary.script.script.Script* attribute), 8
 TAGS (*ieml.constants.DescriptorsType* attribute), 26
 tail (*ieml.usl.decoration.path.UslPath* attribute), 21
 tokens (*ieml.dictionary.script.parser.parser.ScriptParser* attribute), 11
 tokens (*ieml.usl.decoration.parser.parser.PathParser* attribute), 21
 tokens (*ieml.usl.parser.parser.IEMLParser* attribute), 18
 TRANSLATIONS (*ieml.constants.DescriptorsType* attribute), 26
 tree_iter() (*ieml.commons.TreeStructure* method), 25
 TreeStructure (class in *ieml.commons*), 25

U

update() (*ieml.commons.FolderWatcherCache* method), 24
 update_all_ieml() (*ieml.ieml_database.transactions.DBTransaction.DBTransactions* method), 29
 update_morpheme_paradigm() (*ieml.ieml_database.transactions.DBTransaction.DBTransactions* method), 29
 USL (class in *ieml.usl.usl*), 11
 usl() (in module *ieml.usl.usl*), 12
 usl_from_path_values() (in module *ieml.usl.decoration.path*), 21
 USL_TYPE (*ieml.usl.decoration.path.FlexionPath* attribute), 18
 USL_TYPE (*ieml.usl.decoration.path.LexemePath* attribute), 19
 USL_TYPE (*ieml.usl.decoration.path.PolymorphemePath* attribute), 19
 USL_TYPE (*ieml.usl.decoration.path.RolePath* attribute), 20
 USL_TYPE (*ieml.usl.decoration.path.UslPath* attribute), 20
 UslPath (class in *ieml.usl.decoration.path*), 20
 UslTable2D (class in *ieml.usl.table*), 16

V

Variation (class in *ieml.usl.variation*), 17

W

without_morpheme() (*ieml.usl.decoration.path.FlexionPath* method), 18
 without_morpheme() (*ieml.usl.decoration.path.LexemePath* method), 19

```
without_morpheme()
    (ieml.usl.decoration.path.PolymorphemePath
     method), 19
without_morpheme()
    (ieml.usl.decoration.path.RolePath method), 20
without_morpheme()
    (ieml.usl.decoration.path.UslPath method),
    21
Word (class in ieml.usl.word), 15
```