
ConferenceCorpus

Wolfgang Fahl

May 11, 2022

CONTENTS:

1	corpus package	1
1.1	Submodules	1
1.2	corpus.event module	1
1.3	corpus.eventcorpus module	6
1.4	corpus.lookup module	7
1.5	Module contents	9
2	datasources package	11
2.1	Submodules	11
2.2	datasources.confref module	11
2.3	datasources.crossref module	11
2.4	datasources.dblp module	11
2.5	datasources.dblpxml module	11
2.6	datasources.openresearch module	11
2.7	datasources.webscrape module	11
2.8	datasources.wikicfp module	11
2.9	datasources.wikicfp scrape module	11
2.10	datasources.wikidata module	11
2.11	Module contents	11
3	setup module	13
4	smw package	15
4.1	Submodules	15
4.2	smw.topic module	15
4.3	Module contents	15
5	tests package	17
5.1	Submodules	17
5.2	tests.datasourcetoolbox module	17
5.3	tests.testConfref module	18
5.4	tests.testCorpusLookup module	18
5.5	tests.testCrossRef module	19
5.6	tests.testDblpEvents module	19
5.7	tests.testDblpXml module	20
5.8	tests.testOpenResearch module	21
5.9	tests.testSMW module	23
5.10	tests.testStatistics module	23
5.11	tests.testWebScrape module	24
5.12	tests.testWikiCfp module	24

5.13	tests.testWikiCfpScrape module	24
5.14	tests.testWikidata module	25
5.15	Module contents	26
6	Indices and tables	27
	Python Module Index	29
	Index	31

CORPUS PACKAGE

1.1 Submodules

1.2 corpus.event module

Created on 2021-07-26

@author: wf

class `corpus.event.Event`

Bases: `lodstorage.jsonable.JSONAble`

base class for Event entities

Constructor

asWikiMarkup(*series: str, templateParamLookup: dict*) → *str*

Parameters

- **series** (*str*) – the name of the series
- **templateParamLookup** (*dict*) – the mapping of python attributes to Mediawiki template parameters to be used

Returns my WikiMarkup

Return type *str*

getLookupAcronym()

get the lookup acronym of this event e.g. add year information

Returns the acronym to be used for lookup operations

Return type *str*

getRecord()

get my dict elements that are defined in `getSamples`

Returns fields of my `__dict__` which are defined in `getSamples`

Return type *dict*

mapFromDict(*d: dict, maptuples*)

set my attributes from the given dict mapping with the given mapping (key->attr) tuples

Parameters

- **d** (*dict*) – the dictionary to map
- **maptuples** (*list*) – the list of tuples for mapping

```
class corpus.event.EventBaseManager(name, entityName, entityPluralName: str, listName: Optional[str] =  
None, clazz=None, sourceConfig:  
Optional[corpus.config.EventDataSourceConfig] = None,  
primaryKey: Optional[str] = None, config=None,  
handleInvalidListTypes=False, filterInvalidListTypes=False,  
debug=False, profile=True)
```

Bases: lodstorage.entity.EntityManager

common entity Manager for ConferenceCorpus

Constructor

Parameters

- **name** (*string*) – name of this eventManager
- **entityName** (*string*) – entityType to be managed e.g. Country
- **entityPluralName** (*string*) – plural of the the entityType e.g. Countries
- **config** (*StorageConfig*) – the configuration to be used if None a default configuration will be used
- **handleInvalidListTypes** (*bool*) – True if invalidListTypes should be converted or filtered
- **filterInvalidListTypes** (*bool*) – True if invalidListTypes should be deleted
- **debug** (*boolean*) – override debug setting when default of config is used via config=None
- **profile** (*boolean*) – True if profiling/timing information should be shown for long-running operations

asCsv(*separator: str = ',', selectorCallback: Optional[Callable] = None*)

Converts the events to csv format :param separator: character separating the row values :type separator: str
:param selectorCallback: callback functions returning events to be converted to csv. If None all events are converted.

Returns csv string of events

configure()

configure me - abstract method that needs to be overridden

fromCache(*force: bool = False, getListOfDicts=None, append=False, sampleRecordCount=- 1*)

overwritten version of fromCache that calls postProcessEntityList

fromCsv(*csvString, separator: str = ',', overwriteEvents: bool = True, updateEntitiesCallback:*
Optional[Callable] = None)

Parameters

- **csvString** – csvString having all the csv content
- **separator** – the separator of the csv

- **append** – to append to the self object.
- **updateEntitiesCallback** –

Returns: Nothing. The self object is updated

getEventByKey(*keyToSearch*, *keytype*='pageTitle')

getLoDfromEndpoint() → list

get my content from my endpoint

Returns the list of dicts derived from the given SPARQL query

Return type list

postProcessEntityList(*debug*: bool = False)

postProcess my entities

postProcessLodRecords(*listOfDicts*: list, ***kwargs*)

post process the given list of Dicts with raw Events

Parameters **listOfDicts** (*list*) – the list of raw Events to fix

rateAll(*ratingManager*: *corpus.quality.rating.RatingManager*)

rate all events and series based on the given rating Manager

setAllAttr(*listOfDicts*, *attr*, *value*)

set all attribute values of the given attr in the given list of Dict to the given value

updateFromLod(*lod*: list, *overwriteEvents*: bool = True, *updateEntitiesCallback*: Optional[Callable] = None, *restrictToSamples*: bool = True)

Updates the entities from the given LoD. If a entity does not already exist a new one will be added. :param lod: data to update the entities :param overwriteEvents: If False only missing values are added :param updateEntitiesCallback: Callback function that is called on an updated entity :param restrictToSamples: If True only properties that are names in the samples are set. :type restrictToSamples: bool

Returns:

class `corpus.event.EventManager`(*name*: str, *sourceConfig*: Optional[*corpus.config.EventDataSourceConfig*] = None, *clazz*=None, *primaryKey*: Optional[str] = None, *config*: Optional[*lodstorage.storageconfig.StorageConfig*] = None, *debug*=False)

Bases: `corpus.event.EventBaseManager`

Event entity list

constructor

static `asWikiSon`(*eventDicts*)

static `eventDictToWikiSon`(*eventDict*)

getEventsInSeries(*seriesAcronym*)

Return all the events in a given series.

linkSeriesAndEvent(*eventSeriesManager*: `corpus.event.EventSeriesManager`, *seriesKey*: str = 'series')

link Series and Event using the given foreignKey

Parameters

- **seriesKey** (*str*) – the key to be use for lookup
- **eventSeriesManager** (`EventSeriesManager`) –

class corpus.event.EventSeries

Bases: lodstorage.jsonable.JSONable

base class for Event Series entities

Constructor

asWikiMarkup() → str

convert me to wikimarkup

see <https://github.com/WolfgangFahl/ConferenceCorpus/issues/10>

class corpus.event.EventSeriesManager(*name: str, sourceConfig: Optional[corpus.config.EventDataSourceConfig] = None, clazz=None, primaryKey: Optional[str] = None, config: Optional[lodstorage.storageconfig.StorageConfig] = None, debug=False*)

Bases: *corpus.event.EventBaseManager*

Event series list

constructor

class corpus.event.EventStorage

Bases: object

common storage aspects of the EventManager and EventSeriesManager

classmethod **asPlantUml**(*baseEntity='Event', exclude=None*)

return me as a plantUml Diagram markup

classmethod **createLookup**(*column: str, tables: list*)

create a lookup for a column for the given list of tables

Parameters

- **column** (*str*) – the column to create the lookup for
- **tables** (*str*) – the names of the tables to take into account

classmethod **createViews**(*exclude=None, show=False*)

create the general Event views

Parameters

- **exclude** (*list*) – the list of table names to be excluded
- **show** (*bool*) – if True show the DDL

classmethod **getCommonViewDDLs**(*viewNames=['event', 'eventseries'], exclude=None*)

get the SQL DDL for a common view

Returns the SQL DDL CREATE VIEW command

Return type str

classmethod **getDBFile**(*cacheFileName='EventCorpus'*)

get the database file for the given cacheFileName

Parameters **cacheFileName** (*str*) – the name of the cacheFile without suffix

classmethod `getQueryManager(lang='sql', name='queries', debug=False)`

get the query manager for the given language and fileName

Parameters

- **lang** (*str*) – the language of the queries to extract
- **name** (*str*) – the name of the manager containing the query specifications
- **debug** (*bool*) – if True set debugging on

classmethod `getSignatureCache(profile: bool = True, force: bool = False)`

cache the signature Data in a separate SQLite DB

Parameters

- **profile** (*bool*) – if True show profiling information
- **force** (*bool*) – if True force the cache creation

classmethod `getSqlDB()`

get the SQL Database

static `getStorageConfig(debug: bool = False, mode='sql') → lodstorage.storageconfig.StorageConfig`

get the storageConfiguration

Parameters

- **debug** (*bool*) – if True show debug information
- **mode** (*str*) – sql or json

Returns the storage configuration to be used

Return type StorageConfig

classmethod `getTableList(withInstanceCount: bool = True) → list`

get the list of SQL Tables involved

Returns the map of SQL tables used for caching withInstanceCount(bool): if TRUE add the count of instances to the table Map

Return type list

classmethod `getViewTableList(viewName, exclude=None)`

`profile = True`

`viewTableExcludes = {'event': ['event_acm', 'event_ceurws', 'event_orclonebackup', 'event_or', 'event_orbackup'], 'eventseries': ['eventseries_acm', 'eventseries_or', 'eventseries_orbackup', 'eventseries_orclonebackup', 'eventseries_gnd']}`

`withShowProgress = False`

1.3 corpus.eventcorpus module

Created on 2021-04-16

@author: wf

class corpus.eventcorpus.EventCorpus(*debug=False, verbose=False*)

Bases: object

Towards a gold standard event corpus and observatory ...

Constructor

Parameters

- **debug** (*bool*) – set debugging if True
- **verbose** (*bool*) – set verbose output if True

addDataSource(*eventDataSource: corpus.eventcorpus.EventDataSource*)

adds the given eventDataSource

Parameters **eventDataSource** – EventDataSource

static **download**()

download the EventCorpus.db if needed

loadAll(*forceUpdate: bool = False, showProgress=False*)

load all eventDataSources

Parameters **forceUpdate** (*bool*) – True if the data should be fetched from the source instead of the cache

class corpus.eventcorpus.EventDataSource(*eventManager: corpus.event.EventManager, eventSeriesManager: corpus.event.EventSeriesManager, sourceConfig=<class 'corpus.config.EventDataSourceConfig'>*)

Bases: object

a data source for events

constructor

Parameters

- **sourceConfig** (*EventDataSourceConfig*) – the configuration for the EventDataSource
- **eventManager** (*EventManager*) – manager for the events
- **eventSeriesManager** (*EventSeriesManager*) – manager for the eventSeries

load(*forceUpdate=False, showProgress=False, debug=False*)

load this data source

Parameters

- **forceUpdate** (*bool*) – if true force updating this datasource
- **showProgress** (*bool*) – if true show the progress
- **debug** (*bool*) – if true show debug information

rateAll(*ratingManager: corpus.quality.rating.RatingManager*)

rate all events and series based on the given rating Manager

1.4 corpus.lookup module

Created on 2021-07-30

@author: wf

```
class corpus.lookup.CorpusLookup(lookupIds: Optional[list] = None, configure: Optional[callable] = None,
                                debug=False)
```

Bases: object

search and lookup for different EventCorpora

Constructor

Parameters

- **lookupIds** (*list*) – the list of lookupIds to addDataSources for
- **configure** (*callable*) – Callback to configure the corpus lookup

```
getDataSource(lookupId: str) → corpus.eventcorpus.EventDataSource
```

get the data source by the given lookupId

Parameters **lookupId** (*str*) – the lookupId of the data source to get

Returns the data source

Return type *EventDataSource*

```
getDataSource4TableName(tableName: str) → corpus.eventcorpus.EventDataSource
```

get the data source by the given tableName

Parameters **tableName** (*str*) – a tableName of the data source to get

Returns the data source

Return type *EventDataSource*

```
getDataSourceInfos(withInstanceCount: bool = True)
```

get the dataSource Infos

```
getDictOfLod4MultiQuery(multiquery: str, idQuery: Optional[str] = None, omitFailed: bool = True) → dict
```

Parameters

- **multiquery** (*str*) – the multi query containing a variable
- **idQuery** (*str*) – optional query to get lists of ids for selection
- **omitFailed** (*bool*) – if True omit failed queries if False raise Exception on failure

Returns the dict of list of dicts for the queries derived from the multi query

Return type dict

Raises **Exception** – if omitFailed is False and an error occurred for a query

```
getLod4Query(query: str, params=None)
```

Parameters

- **query** – the query to run
- **params** (*tuple*) – the query params, if any

Returns the list of dicts for the query

Return type list

getMultiQueryVariable(*multiquery: str, lenient: bool = False*)

get the variable being used in a multiquery

Parameters

- **multiquery** (*str*) – the multiquery containing a {variable}
- **lenient** (*bool*) – if True allow to return a None value otherwise raise an Exception if no variable was found

Returns variable

Return type str

Raises **Exception** – if lenient is False and no variable was found

load(*forceUpdate: bool = False, showProgress: bool = False, withCreateViews=True*)

load the event corpora

Parameters

- **forceUpdate** (*bool*) – if True the data should be fetched from the source instead of the cache
- **showProgress** (*bool*) – if True the progress of the loading should be shown
- **withCreateViews** (*bool*) – if True recreate the common views

`lookupIds = ['confref', 'crossref', 'dblp', 'gnd', 'tibkat', 'wikidata', 'wikicfp', 'or', 'or-backup', 'orclone', 'orclone-backup']`

class corpus.lookup.CorporaLookupConfigure

Bases: object

Configuration of the Corpus Lookup

static **configureCorpusLookup**(*lookup, debug=False*)

callback to configure the corpus lookup

static **getWikiTextPath**(*wikiId: str*)

get the WikiText (Backup) path for the given wikiId

Parameters **wikiId** (*str*) – the wikiId (e.g. “or”, “orclone”)

Returns the path to the backup files as created by the wikibackup script

`corpus.lookup.main`(*argv=None*)

main program.

1.5 Module contents

DATASOURCES PACKAGE

2.1 Submodules

2.2 `datasources.confref` module

2.3 `datasources.crossref` module

2.4 `datasources.dblp` module

2.5 `datasources.dblpxml` module

2.6 `datasources.openresearch` module

2.7 `datasources.webscrape` module

2.8 `datasources.wikicfp` module

2.9 `datasources.wikicfp scrape` module

2.10 `datasources.wikidata` module

2.11 Module contents

SETUP MODULE

SMW PACKAGE

4.1 Submodules

4.2 smw.topic module

4.3 Module contents

TESTS PACKAGE

5.1 Submodules

5.2 tests.datasourcetoolbox module

Created on 2021-07-29

@author: wf

class tests.datasourcetoolbox.DataSourceTest(*methodName='runTest'*)

Bases: unittest.case.TestCase

test for EventDataSources

Create an instance of the class that will use the named test method when executed. Raises a ValueError if the instance does not have a method with the specified name.

checkDataSource(*eventDataSource: corpus.eventcorpus.EventDataSource, expectedSeries: int, expectedEvents: int, eventSample: Optional[str] = None*)

check the given DataSource

Parameters **eventDataSource** (**EventDataSource**) – the event data source to check

static **getEventSeries**(*seriesAcronym: str*)

Returns the event series as dict of lod (records are categorized into the different data sources)

Parameters **seriesAcronym** – acronym of the series

Returns dict of lod

inCI()

are we running in a Continuous Integration Environment?

classmethod **main**()

classmethod **optionalDebug**(*args*)

start the remote debugger if the arguments specify so

Parameters **args**() – The command line arguments

setUp(*debug=False, profile=True, timeLimitPerTest=10.0*)

setUp test environment

tearDown()

Hook method for deconstructing the test fixture after testing it.

5.3 tests.testConfref module

Created on 2021-08-02

@author: wf

class tests.testConfref.TestConfRef(*methodName='runTest'*)

Bases: `tests.datasourcetoolbox.DataSourceTest`

test getting events from Confref <http://portal.confref.org> as a data source

Create an instance of the class that will use the named test method when executed. Raises a ValueError if the instance does not have a method with the specified name.

setUp()

setUp test environment

testConfRef()

test CrossRef as an event data source

5.4 tests.testCorpusLookup module

Created on 2021-07-26

@author: wf

class tests.testCorpusLookup.TestCorpusLookup(*methodName='runTest'*)

Bases: `tests.datasourcetoolbox.DataSourceTest`

test the event corpus

Create an instance of the class that will use the named test method when executed. Raises a ValueError if the instance does not have a method with the specified name.

configureCorpusLookup(*lookup: corpus.lookup.CorpusLookup*)

callback to configure the corpus lookup

setUp(*debug=False, profile=True, **kwargs*)

setUp test environment

testDataSource4Table()

test getting datasources by table name

testGetDataSourceInfos()

test getting the infos for the datasources

testLookup()

test the lookup

testMultiQuery()

test getting entries for a given query

testViewDDL()

test the view DDL

5.5 tests.testCrossRef module

Created on 2021-08-02

@author: wf

class tests.testCrossRef.**TestCrossRef**(methodName='runTest')

Bases: `tests.datasourcetoolbox.DataSourceTest`

test getting events from CrossRef <https://www.crossref.org/> as a data source

Create an instance of the class that will use the named test method when executed. Raises a ValueError if the instance does not have a method with the specified name.

prettyJson(d)

pretty print the given jsonStr

setUp()

setUp the test environment

testCrossRef()

test CrossRef as an event data source

testCrossRefEventFromDOI()

test creating CrossRefEvents via API

testCrossref_DOI_Lookup()

test crossref API access see <https://github.com/WolfgangFahl/ProceedingsTitleParser/issues/28>

testFixUmlauts()

workaround Umlaut issue see <https://stackoverflow.com/questions/63486767/how-can-i-get-the-fuseki-api-via-sparqlwrapper-to-properly-report-a-detailed-err>

5.6 tests.testDblpEvents module

Created on 28.07.2021

@author: wf

class tests.testDblpEvents.**TestDblpEvents**(methodName='runTest')

Bases: `tests.datasourcetoolbox.DataSourceTest`

test the dblp data source

Create an instance of the class that will use the named test method when executed. Raises a ValueError if the instance does not have a method with the specified name.

classmethod configureCorpusLookup(lookup: `corpus.lookup.CorporaLookup`)

callback to configure the corpus lookup

setUp(**kwargs)

setup

classmethod setUpClass()

Hook method for setting up class fixture before running tests in the class.

testDateRange()

test date Range parsing

testDblp()

test getting the conference series and events from dblp xml dump

testDblpDateFix()

test Dblp DateRange extraction/fixing

5.7 tests.testDblpXml module

Created on 2021-01-25

@author: wf

class tests.testDblpXml.**TestDblp**(methodName='runTest')

Bases: *tests.datasourcetoolbox.DataSourceTest*

test the dblp xml parser and pylodstorage extraction for it

Create an instance of the class that will use the named test method when executed. Raises a ValueError if the instance does not have a method with the specified name.

checkConfColumn(sqlDB)

check the conference columns

static **getMockedDblp**(mock=True, debug=False)

getSqlDB(mock=True, recreate=False)

get the Sql Database

log(msg)

mock = **True**

setUp(debug: bool = False, profile: bool = True, **kwargs)

setUp the test environment

especially the mocking parameter - if mock is False a multi-Gigabyte download might be activated

tearDown()

Hook method for deconstructing the test fixture after testing it.

testCreateSample()

test creating a sample file

testDblpDownload()

test dblp access

testDblpXmlParser()

test parsing the xml file

testIssue5()

<https://github.com/WolfgangFahl/ConferenceCorpus/issues/5>

dblp xml parser skips some proceedings titles

testQueries()

test the parameterized query

testSqlLiteDatabaseCreation()

get dict of list of dicts (tables)

testUml()

test generating the uml diagram for the entities

5.8 tests.testOpenResearch module

Created on 27.07.2021

@author: wf

class tests.testOpenResearch.TestOREEventManager(*methodName='runTest'*)

Bases: [tests.datasourcetoolbox.DataSourceTest](#)

Tests OREEventManager

Create an instance of the class that will use the named test method when executed. Raises a ValueError if the instance does not have a method with the specified name.

setUp(*debug=False, profile=True, timeLimitPerTest=10.0*)

setUp test environment

test_configure()

tests configuring OREEventManager with different loading methods to retrieve the records from the source

test_fromWikiFileManager()

tests fromWikiFileManager

test_fromWikiUser()

tests fromWikiUser

test_getLodFrom()

tests the different loading methods Note: the functionality is tested in depth in TestOrSMW here only enhancing of the records is tested

test_getPropertyLookup()

tests getPropertyLookup

class tests.testOpenResearch.TestOREEventSeriesManager(*methodName='runTest'*)

Bases: [tests.datasourcetoolbox.DataSourceTest](#)

Tests OREEventSeriesManager

Create an instance of the class that will use the named test method when executed. Raises a ValueError if the instance does not have a method with the specified name.

setUp(*debug=False, profile=True, timeLimitPerTest=10.0*)

setUp test environment

test_configure()

tests configuring OREEventManager with different loading methods to retrieve the records from the source

test_fromWikiFileManager()

tests fromWikiFileManager

test_fromWikiUser()

tests fromWikiUser

test_getLodFrom()

tests the different loading methods Note: the functionality is tested in depth in TestOrSMW here only the enhancing of the records is tested

test_getPropertyLookup()

tests getPropertyLookup

class tests.testOpenResearch.TestOpenResearch(*methodName='runTest'*)

Bases: [tests.datasourcetoolbox.DataSourceTest](#)

test the access to OpenResearch

Create an instance of the class that will use the named test method when executed. Raises a ValueError if the instance does not have a method with the specified name.

setUp(*debug=False, profile=True, **kwargs*)

setUp test environment

testAsCsv()

test csv export of events

testORDataSourceFromWikiFileManager()

tests the getting conferences form wiki markup files

testORDataSourceFromWikiUser()

tests initializing the OREventCorpus from wiki

class tests.testOpenResearch.TestOrSMW(*methodName='runTest'*)

Bases: [tests.basetest.BaseTest](#)

tests OrSWM

Create an instance of the class that will use the named test method when executed. Raises a ValueError if the instance does not have a method with the specified name.

setUp(*debug: bool = False, profile: bool = True*)

setUp test environment

test_getAskQuery()

tests getAskQuery

test_getAskQueryPageTitles()

tests getAskQueryPageTitles

test_getLodFromWikiApi()

tests getLodFromWikiApi

test_getLodFromWikiFiles()

tests getLodFromWikiFiles

test_getLodFromWikiMarkup()

tests getLodFromWikiMarkup

test_getLodFromWikiMarkup_queryAllEvents()

tests querying all events with getLodFromWikiApi to check if the query division works

test_normalizeProperties()

tests normalizeProperties

5.9 tests.testSMW module

class tests.testSMW.TestSMW

Bases: object

tests functionalities of the smw package provides helper functions for other tests

classmethod getSMW_WikiUser(*wikiId='or', save=False*)

get semantic media wiki users for SemanticMediawiki.org and openresearch.org

classmethod getWikiFileManager(*wikiId=None, debug=False*)

classmethod getWikiUser(*wikiId=None*)

static inPublicCI()

are we running in a public Continuous Integration Environment?

wikiId = 'orclone'

5.10 tests.testStatistics module

Created on 2021-07-31

@author: wf

class tests.testStatistics.TestStatistics(*methodName='runTest'*)

Bases: *tests.datasourcetoolbox.DataSourceTest*

test statistics

Create an instance of the class that will use the named test method when executed. Raises a ValueError if the instance does not have a method with the specified name.

setUp(*debug: bool = False, profile: bool = True, **kwargs*)

setUp test environment

classmethod setUpClass()

Hook method for setting up class fixture before running tests in the class.

testIssue255()

test issue 255 problem list

testMultiQuery()

test MultiQuery

testStatistics()

test statistics

5.11 tests.testWebScrape module

Created on 2021-07-31

@author: wf

class tests.testWebScrape.TestWebScrape(*methodName='runTest'*)

Bases: [tests.datasourcetoolbox.DataSourceTest](#)

test getting rdfA based triples from Webpages

Create an instance of the class that will use the named test method when executed. Raises a ValueError if the instance does not have a method with the specified name.

testCrawlType()

test CrawlType is Valid

testWebScrape()

test getting rdfA encoded info from a webpage

5.12 tests.testWikiCfp module

Created on 2021-07-31

@author: wf

class tests.testWikiCfp.TestWikiCFP(*methodName='runTest'*)

Bases: [tests.datasourcetoolbox.DataSourceTest](#)

test WikiCFP data source

Create an instance of the class that will use the named test method when executed. Raises a ValueError if the instance does not have a method with the specified name.

setUp(*debug: bool = False, profile: bool = True, **kwargs*)

setUp test environment

testWikiCFP()

test the WikiCFP lookup

5.13 tests.testWikiCfpScrape module

Created on 2020-08-20

@author: wf

class tests.testWikiCfpScrape.TestWikiCFP(*methodName='runTest'*)

Bases: [tests.datasourcetoolbox.DataSourceTest](#)

test events from WikiCFP

Create an instance of the class that will use the named test method when executed. Raises a ValueError if the instance does not have a method with the specified name.

getTempJsonDir() → str

handleError(*ex*)

handle the given exception

Parameters **ex** (*Exception*) – the exception to handle

printDelimiterCount(*names*)

print the count of the most common used delimiters in the given name list

setUp(*debug: bool = False, profile: bool = True, **kwargs*)

setUp test environment

testCrawlEvents()

test crawling a few events and storing the result to a json file

testCrawlEventsViaCommandLine()

test crawling via commandline

testCrawlFilesToJson()

test getting the crawlFiles content

testCrawlType()

test CrawlType enumeration

testCrawledJsonFiles()

get the crawl files

testEventScraping()

test scraping the given event

test “This item has been deleted” WikiCFP items

e.g. <http://www.wikicfp.com/cfp/servlet/event.showcfp?eventid=3>

testGettingEventSeriesForEvent()

test extracting the event series id from th event page

testGettingLatestEvent()

get the latest event Id with a binary search

testInvalidUrl()

make sure only valid urls are accepted

testJsonPickleDateTime()

test the JsonPickle datetime encoding mystery

testWikiCFP()

test event handling from WikiCFP

5.14 tests.testWikidata module

Created on 27.07.2021

@author: wf

class tests.testWikidata.**TestWikiData**(*methodName='runTest'*)

Bases: `tests.datasourcetoolbox.DataSourceTest`

test wiki data access

Create an instance of the class that will use the named test method when executed. Raises a ValueError if the instance does not have a method with the specified name.

setUp(*debug: bool = False, profile: bool = True, **kwargs*)

setUp test environment

testQueryManager()

test named query usage see <https://github.com/WolfgangFahl/ConferenceCorpus/issues/45>

testRegExp()

test Regexp handling for wikiedit

testWikidata()

test getting the wikiData Event Series

5.15 Module contents

INDICES AND TABLES

- `genindex`
- `modindex`
- `search`

PYTHON MODULE INDEX

C

`corpus`, 9
`corpus.event`, 1
`corpus.eventcorpus`, 6
`corpus.lookup`, 7

t

`tests`, 26
`tests.datasourcetoolbox`, 17
`tests.testConfref`, 18
`tests.testCorpusLookup`, 18
`tests.testCrossRef`, 19
`tests.testDblpEvents`, 19
`tests.testDblpXml`, 20
`tests.testOpenResearch`, 21
`tests.testSMW`, 23
`tests.testStatistics`, 23
`tests.testWebScrape`, 24
`tests.testWikiCfp`, 24
`tests.testWikiCfpScrape`, 24
`tests.testWikidata`, 25

A

addDataSource() (*corpus.eventcorpus.EventCorpus* method), 6
 asCsv() (*corpus.event.EventBaseManager* method), 2
 asPlantUml() (*corpus.event.EventStorage* class method), 4
 asWikiMarkup() (*corpus.event.Event* method), 1
 asWikiMarkup() (*corpus.event.EventSeries* method), 4
 asWikiSon() (*corpus.event.EventManager* static method), 3

C

checkConfColumn() (*tests.testDbpXml.TestDbp* method), 20
 checkDataSource() (*tests.datasourcetoolbox.DataSource* method), 17
 configure() (*corpus.event.EventBaseManager* method), 2
 configureCorpusLookup() (*corpus.lookup.CorporisLookupConfigure* static method), 8
 configureCorpusLookup() (*tests.testCorpusLookup.TestCorpusLookup* method), 18
 configureCorpusLookup() (*tests.testDbpEvents.TestDbpEvents* class method), 19
 corpus
 module, 9
 corpus.event
 module, 1
 corpus.eventcorpus
 module, 6
 corpus.lookup
 module, 7
 CorpusLookup (class in *corpus.lookup*), 7
 CorpusLookupConfigure (class in *corpus.lookup*), 8
 createLookup() (*corpus.event.EventStorage* class method), 4
 createViews() (*corpus.event.EventStorage* class method), 4

D

DataSourceTest (class in *tests.datasourcetoolbox*), 17
 download() (*corpus.eventcorpus.EventCorpus* static method), 6

E

Event (class in *corpus.event*), 1
 EventBaseManager (class in *corpus.event*), 2
 EventCorpus (class in *corpus.eventcorpus*), 6
 EventDataSource (class in *corpus.eventcorpus*), 6
 eventDictToWikiSon() (*corpus.event.EventManager* static method), 3
 EventManager (class in *corpus.event*), 3
 EventSeries (class in *corpus.event*), 3
 EventSeriesManager (class in *corpus.event*), 4
 EventStorage (class in *corpus.event*), 4

F

fromCache() (*corpus.event.EventBaseManager* method), 2
 fromCsv() (*corpus.event.EventBaseManager* method), 2

G

getCommonViewDDLs() (*corpus.event.EventStorage* class method), 4
 getDataSource() (*corpus.lookup.CorporisLookup* method), 7
 getDataSource4TableName() (*corpus.lookup.CorporisLookup* method), 7
 getDataSourceInfos() (*corpus.lookup.CorporisLookup* method), 7
 getDBFile() (*corpus.event.EventStorage* class method), 4
 getDictOfLod4MultiQuery() (*corpus.lookup.CorporisLookup* method), 7
 getEventByKey() (*corpus.event.EventBaseManager* method), 3
 getEventSeries() (*tests.datasourcetoolbox.DataSourceTest* static method), 17
 getEventsInSeries() (*corpus.event.EventManager* method), 3

`getLod4Query()` (*corpus.lookup.CorporaLookup method*), 7
`getLodFromEndpoint()` (*corpus.event.EventBaseManager method*), 3
`getLookupAcronym()` (*corpus.event.Event method*), 1
`getMockedDbp()` (*tests.testDbpXml.TestDbp static method*), 20
`getMultiQueryVariable()` (*corpus.lookup.CorporaLookup method*), 8
`getQueryManager()` (*corpus.event.EventStorage class method*), 4
`getRecord()` (*corpus.event.Event method*), 1
`getSignatureCache()` (*corpus.event.EventStorage class method*), 5
`getSMW_WikiUser()` (*tests.testSMW.TestSMW class method*), 23
`getSqlDB()` (*corpus.event.EventStorage class method*), 5
`getSqlDB()` (*tests.testDbpXml.TestDbp method*), 20
`getStorageConfig()` (*corpus.event.EventStorage static method*), 5
`getTableList()` (*corpus.event.EventStorage class method*), 5
`getTempJsonDir()` (*tests.testWikiCfpScrape.TestWikiCFP method*), 24
`getViewTableList()` (*corpus.event.EventStorage class method*), 5
`getWikiFileManager()` (*tests.testSMW.TestSMW class method*), 23
`getWikiTextPath()` (*corpus.lookup.CorporaLookupConfigure static method*), 8
`getWikiUser()` (*tests.testSMW.TestSMW class method*), 23

H

`handleError()` (*tests.testWikiCfpScrape.TestWikiCFP method*), 24

I

`inCI()` (*tests.datasourcetoolbox.DataSourceTest method*), 17
`inPublicCI()` (*tests.testSMW.TestSMW static method*), 23

L

`linkSeriesAndEvent()` (*corpus.event.EventManager method*), 3
`load()` (*corpus.eventcorpus.EventDataSource method*), 6
`load()` (*corpus.lookup.CorporaLookup method*), 8
`loadAll()` (*corpus.eventcorpus.EventCorpus method*), 6
`log()` (*tests.testDbpXml.TestDbp method*), 20
`lookupIds` (*corpus.lookup.CorporaLookup attribute*), 8

M

`main()` (*in module corpus.lookup*), 8
`main()` (*tests.datasourcetoolbox.DataSourceTest class method*), 17
`mapFromDict()` (*corpus.event.Event method*), 1
`mock` (*tests.testDbpXml.TestDbp attribute*), 20
`module`
 corpus, 9
 corpus.event, 1
 corpus.eventcorpus, 6
 corpus.lookup, 7
 tests, 26
 tests.datasourcetoolbox, 17
 tests.testConfref, 18
 tests.testCorpusLookup, 18
 tests.testCrossRef, 19
 tests.testDbpEvents, 19
 tests.testDbpXml, 20
 tests.testOpenResearch, 21
 tests.testSMW, 23
 tests.testStatistics, 23
 tests.testWebScrape, 24
 tests.testWikiCfp, 24
 tests.testWikiCfpScrape, 24
 tests.testWikidata, 25

O

`optionalDebug()` (*tests.datasourcetoolbox.DataSourceTest class method*), 17

P

`postProcessEntityList()` (*corpus.event.EventBaseManager method*), 3
`postProcessLodRecords()` (*corpus.event.EventBaseManager method*), 3
`prettyJson()` (*tests.testCrossRef.TestCrossRef method*), 19
`printDelimiterCount()` (*tests.testWikiCfpScrape.TestWikiCFP method*), 25
`profile` (*corpus.event.EventStorage attribute*), 5

R

`rateAll()` (*corpus.event.EventBaseManager method*), 3
`rateAll()` (*corpus.eventcorpus.EventDataSource method*), 6

S

`setAllAttr()` (*corpus.event.EventBaseManager method*), 3
`setUp()` (*tests.datasourcetoolbox.DataSourceTest method*), 17
`setUp()` (*tests.testConfref.TestConfRef method*), 18

setUp() (tests.testCorpusLookup.TestCorpusLookup method), 18
 setUp() (tests.testCrossRef.TestCrossRef method), 19
 setUp() (tests.testDblpEvents.TestDblpEvents method), 19
 setUp() (tests.testDblpXml.TestDblp method), 20
 setUp() (tests.testOpenResearch.TestOpenResearch method), 22
 setUp() (tests.testOpenResearch.TestOREEventManager method), 21
 setUp() (tests.testOpenResearch.TestOREEventSeriesManager method), 21
 setUp() (tests.testOpenResearch.TestOrSMW method), 22
 setUp() (tests.testStatistics.TestStatistics method), 23
 setUp() (tests.testWikiCfp.TestWikiCFP method), 24
 setUp() (tests.testWikiCfpScrape.TestWikiCFP method), 25
 setUp() (tests.testWikidata.TestWikiData method), 26
 setUpClass() (tests.testDblpEvents.TestDblpEvents class method), 19
 setUpClass() (tests.testStatistics.TestStatistics class method), 23

T
 tearDown() (tests.datasourcetoolbox.DataSourceTest method), 17
 tearDown() (tests.testDblpXml.TestDblp method), 20
 test_configure() (tests.testOpenResearch.TestOREEventManager method), 21
 test_configure() (tests.testOpenResearch.TestOREEventSeriesManager method), 21
 test_fromWikiFileManager() (tests.testOpenResearch.TestOREEventManager method), 21
 test_fromWikiFileManager() (tests.testOpenResearch.TestOREEventSeriesManager method), 21
 test_fromWikiUser() (tests.testOpenResearch.TestOREEventManager method), 21
 test_fromWikiUser() (tests.testOpenResearch.TestOREEventSeriesManager method), 21
 test_getAskQuery() (tests.testOpenResearch.TestOrSMW method), 22
 test_getAskQueryPageTitles() (tests.testOpenResearch.TestOrSMW method), 22
 test_getLodFrom() (tests.testOpenResearch.TestOREEventManager method), 21
 test_getLodFrom() (tests.testOpenResearch.TestOREEventSeriesManager method), 22
 test_getLodFromWikiApi() (tests.testOpenResearch.TestOrSMW method), 22
 test_getLodFromWikiFiles() (tests.testOpenResearch.TestOrSMW method), 22
 test_getLodFromWikiMarkup() (tests.testOpenResearch.TestOrSMW method), 22
 test_getLodFromWikiMarkup_queryAllEvents() (tests.testOpenResearch.TestOrSMW method), 22
 test_getPropertyLookup() (tests.testOpenResearch.TestOREEventManager method), 21
 test_getPropertyLookup() (tests.testOpenResearch.TestOREEventSeriesManager method), 22
 test_normalizeProperties() (tests.testOpenResearch.TestOrSMW method), 22
 testAsCsv() (tests.testOpenResearch.TestOpenResearch method), 22
 TestConfRef (class in tests.testConfref), 18
 testConfRef() (tests.testConfref.TestConfRef method), 18
 TestCorpusLookup (class in tests.testCorpusLookup), 18
 testCrawledJsonFiles() (tests.testWikiCfpScrape.TestWikiCFP method), 25
 testCrawlEvents() (tests.testWikiCfpScrape.TestWikiCFP method), 25
 testCrawlEventsViaCommandLine() (tests.testWikiCfpScrape.TestWikiCFP method), 25
 testCrawlFilesToJson() (tests.testWikiCfpScrape.TestWikiCFP method), 25
 testCrawlType() (tests.testWebScrape.TestWebScrape method), 24
 testCrawlType() (tests.testWikiCfpScrape.TestWikiCFP method), 25
 testCreateSample() (tests.testDblpXml.TestDblp method), 20
 TestCrossRef (class in tests.testCrossRef), 19
 testCrossRef() (tests.testCrossRef.TestCrossRef method), 19
 testCrossref_DOI_Lookup() (tests.testCrossRef.TestCrossRef method), 19
 testCrossRefEventFromDOI() (tests.testCrossRef.TestCrossRef method), 19

testDataSource4Table() (tests.testCorpusLookup.TestCorpusLookup method), 18
 testDateRange() (tests.testDbpEvents.TestDbpEvents method), 19
 TestDbp (class in tests.testDbpXml), 20
 testDbp() (tests.testDbpEvents.TestDbpEvents method), 20
 testDbpDateFix() (tests.testDbpEvents.TestDbpEvents method), 20
 testDbpDownload() (tests.testDbpXml.TestDbp method), 20
 TestDbpEvents (class in tests.testDbpEvents), 19
 testDbpXmlParser() (tests.testDbpXml.TestDbp method), 20
 testEventScraping() (tests.testWikiCfpScrape.TestWikiCFP method), 25
 testFixUmlauts() (tests.testCrossRef.TestCrossRef method), 19
 testGetDataSourceInfos() (tests.testCorpusLookup.TestCorpusLookup method), 18
 testGettingEventSeriesForEvent() (tests.testWikiCfpScrape.TestWikiCFP method), 25
 testGettingLatestEvent() (tests.testWikiCfpScrape.TestWikiCFP method), 25
 testInvalidUrl() (tests.testWikiCfpScrape.TestWikiCFP method), 25
 testIssue255() (tests.testStatistics.TestStatistics method), 23
 testIssue5() (tests.testDbpXml.TestDbp method), 20
 testJsonPickleDateTime() (tests.testWikiCfpScrape.TestWikiCFP method), 25
 testLookup() (tests.testCorpusLookup.TestCorpusLookup method), 18
 testMultiQuery() (tests.testCorpusLookup.TestCorpusLookup method), 18
 testMultiQuery() (tests.testStatistics.TestStatistics method), 23
 TestOpenResearch (class in tests.testOpenResearch), 22
 testORDataSourceFromWikiFileManager() (tests.testOpenResearch.TestOpenResearch method), 22
 testORDataSourceFromWikiUser() (tests.testOpenResearch.TestOpenResearch method), 22
 TestOREEventManager (class in tests.testOpenResearch), 21
 TestOREEventSeriesManager (class in tests.testOpenResearch), 21
 TestOrSMW (class in tests.testOpenResearch), 22
 testQueries() (tests.testDbpXml.TestDbp method), 20
 testQueryManager() (tests.testWikidata.TestWikiData method), 26
 testRegExp() (tests.testWikidata.TestWikiData method), 26
 tests
 module, 26
 tests.datasourcetoolbox
 module, 17
 tests.testConfref
 module, 18
 tests.testCorpusLookup
 module, 18
 tests.testCrossRef
 module, 19
 tests.testDbpEvents
 module, 19
 tests.testDbpXml
 module, 20
 tests.testOpenResearch
 module, 21
 tests.testSMW
 module, 23
 tests.testStatistics
 module, 23
 tests.testWebScrape
 module, 24
 tests.testWikiCfp
 module, 24
 tests.testWikiCfpScrape
 module, 24
 tests.testWikidata
 module, 25
 TestSMW (class in tests.testSMW), 23
 testSqlLiteDatabaseCreation() (tests.testDbpXml.TestDbp method), 21
 TestStatistics (class in tests.testStatistics), 23
 testStatistics() (tests.testStatistics.TestStatistics method), 23
 testUml() (tests.testDbpXml.TestDbp method), 21
 testViewDDL() (tests.testCorpusLookup.TestCorpusLookup method), 18
 TestWebScrape (class in tests.testWebScrape), 24
 testWebScrape() (tests.testWebScrape.TestWebScrape method), 24
 TestWikiCFP (class in tests.testWikiCfp), 24
 TestWikiCFP (class in tests.testWikiCfpScrape), 24
 testWikiCFP() (tests.testWikiCfp.TestWikiCFP method), 24
 testWikiCFP() (tests.testWikiCfpScrape.TestWikiCFP method), 25
 TestWikiData (class in tests.testWikidata), 25

`testWikidata()` (*tests.testWikidata.TestWikiData*
method), 26

U

`updateFromLod()` (*corpus.event.EventBaseManager*
method), 3

V

`viewTableExcludes` (*corpus.event.EventStorage* *at-*
tribute), 5

W

`wikiId` (*tests.testSMW.TestSMW attribute*), 23

`withShowProgress` (*corpus.event.EventStorage* *at-*
tribute), 5