News Article Extractor

Generated by Doxygen 1.8.17

Chapter 1

NewsArticleExtractor

Website news article scraper and geospatial enabler

THIS IS A WORK IN PROGRESS!!!

Am moving from my personal repository to Github. It likely will not run currently and I also need to upload the SQL file you can use to create the DB.

Chapter 2

Namespace Index

2.1 Namespace List

Here is a list of all namespaces with brief descriptions:

Chapter 3

Hierarchical Index

3.1 Class Hierarchy

This inheritance list is sorted roughly, but not completely, alphabetically:

Chapter 4

Class Index

4.1 Class List

Here are the classes, structs, unions and interfaces with brief descriptions:

Chapter 5

File Index

5.1 File List

Here is a list of all files with brief descriptions:

Chapter 6

Namespace Documentation

6.1 DBWriter Namespace Reference

Namespaces

- DBWriter

6.2 DBWriter.DBWriter Namespace Reference

Classes

- class DBWriter

6.3 newsextractor Namespace Reference

Functions

- List GetSubscriptions (ConfigParser inConfigObject)
- def AddPoisonPill ()

Variables

- localtime = time.asctime(time.localtime(time.time()))
- articleQueue = multiprocessing.Queue()
- plugin_path = os.path.dirname(os.path.realpath(__file__))
- configObject = ConfigParser()
- List urlList = GetSubscriptions(configObject)
- dbWriter = DBWriter(articleQueue, configObject)
- websiteObject = websites.WebsiteFactory(url, articleQueue, configObject)

6.3.1 Function Documentation

6.3.1.1 AddPoisonPill()

```
def newsextractor.AddPoisonPill ( )
```

Adds the poison pill to the queue so we shut down gracefully
:return: Nothing

Definition at line 63 of file newsextractor.py.

```
63 def AddPoisonPill():
64     """
65     Adds the poison pill to the queue so we shut down gracefully
66     :return: Nothing
67     """
68 
69     try:
70         # Shut down the dbwriter
71         killArticle = websites.ExtractedArticle()
72 
73         # Create the poison pill
74         killArticle.articleText = "EXITCALLED"
75 
76         # Place the poison pill on the queue
77         articleQueue.put(killArticle)
78 
79         return
80     except Exception as e:
81         print("newsextractor: Exception in AddPoisonPill: {}".format(e))
82         return
83 
84 
85 #
       ********************************************************************************************************************************************
```

6.3.1.2 GetSubscriptions()

```
List newsextractor.GetSubscriptions (
             ConfigParser inConfigObject )
```

Gets the list of subscripts from the database
:return: List of tuples

Definition at line 20 of file newsextractor.py.

```
20 def GetSubscriptions(inConfigObject: ConfigParser) -> List:
21     """
22     Gets the list of subscripts from the database
23     :return: List of tuples
24     """
25 
26     try:
27         returnList = list()
28 
29         # Get the top level
30         DB = inConfigObject["DB"]
31 
32         # Now set our member variables
33         DBHost = DB["DBHost"]
34         DBPort = DB["DBPort"]
35         DBUser = DB["DBUser"]
36         DBPassword = DB["DBPassword"]
37         DBTable = DB["DBTable"]
38 
```

```
39          DBCOnnection = psycopg2.connect(host=DBHost,
40                                          port=DBPort,
41                                          dbname=DBTable,
42                                          user=DBUser,
43                                          password=DBPassword)
44          cursor = DBCOnnection.cursor()
45
46          cursor.execute("SELECT url, classname FROM subscriptions;")
47          DBCOnnection.commit()
48
49          returnList = cursor.fetchall()
50
51          # clean up
52          cursor.close()
53          DBCOnnection.close()
54
55          return returnList
56
57      except Exception as gsException:
58          print("Exception in newsextractor::GetSubscriptions: {}".format(gsException))
59          return list()
60
61
62  #
      ***************************************************************************************************************************************************
```

6.3.2 Variable Documentation

6.3.2.1 articleQueue

```
newsextractor.articleQueue = multiprocessing.Queue()
```

Definition at line 92 of file newsextractor.py.

6.3.2.2 configObject

```
newsextractor.configObject = ConfigParser()
```

Definition at line 99 of file newsextractor.py.

6.3.2.3 dbWriter

```
newsextractor.dbWriter = DBWriter(articleQueue, configObject)
```

Definition at line 111 of file newsextractor.py.

6.3.2.4 localtime

```
newsextractor.localtime = time.asctime(time.localtime(time.time()))
```

Definition at line 88 of file newsextractor.py.

6.3.2.5 plugin_path

```
newsextractor.plugin_path = os.path.dirname(os.path.realpath(__file__))
```

Definition at line 96 of file newsextractor.py.

6.3.2.6 urlList

```
List newsextractor.urlList = GetSubscriptions(configObject)
```

Definition at line 103 of file newsextractor.py.

6.3.2.7 websiteObject

```
newsextractor.websiteObject = websites.WebsiteFactory(url, articleQueue, configObject)
```

Definition at line 116 of file newsextractor.py.

6.4 websites Namespace Reference

Namespaces

- BBCWebsite
- ExtractedArticle
- WebsiteBase

Functions

- WebsiteBase WebsiteFactory (tuple inURL, multiprocessing.Queue inQueue, configparser.ConfigParser in↩ ConfigObject)

6.4.1 Function Documentation

6.4.1.1 WebsiteFactory()

```
WebsiteBase websites.WebsiteFactory (
             tuple inURL,
             multiprocessing.Queue inQueue,
             configparser.ConfigParser inConfigObject )
```

Factory class method to load and return a module that handles the passed-in string

Definition at line 9 of file __init__.py.

```
9 def WebsiteFactory(inURL: tuple, inQueue: multiprocessing.Queue, inConfigObject:
      configparser.ConfigParser) -> WebsiteBase:
10     """
11     Factory class method to load and return a module that handles the passed-in string
12     """
13
14     if not inURL or not inQueue or not inConfigObject:
15         return None
16
17     # Instance object to return
18     instance = None
19
20     try:
21         # Pull the URL and class name from the tuple
22         url, classname = inURL
23
24         # Load the module
25         classmodule = import_module("." + classname, package="websites")
26
27         tempclass = getattr(classmodule, classname)
28
29         instance = tempclass(inQueue, inConfigObject, url)
30
31         return instance
32     except Exception as e:
33         raise ImportError("The URL {} cannot be handled at this time.".format(inURL))
```

6.5 websites.BBCWebsite Namespace Reference

Classes

- class BBCWebsite

6.6 websites.ExtractedArticle Namespace Reference

Classes

- class ExtractedArticle

6.7 websites.WebsiteBase Namespace Reference

Classes

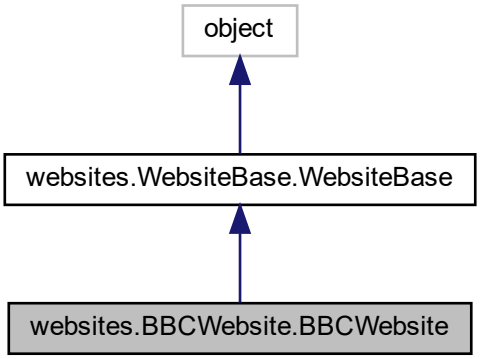
- class WebsiteBase

Chapter 7

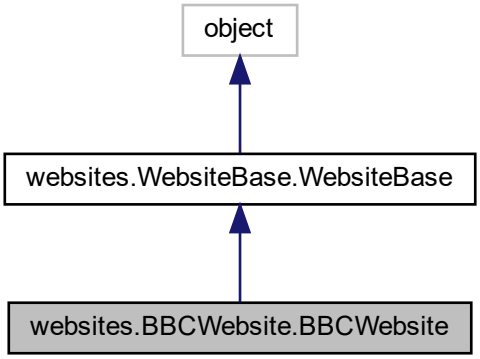
Class Documentation

7.1 websites.BBCWebsite.BBCWebsite Class Reference

Inheritance diagram for websites.BBCWebsite.BBCWebsite:



Collaboration diagram for websites.BBCWebsite.BBCWebsite:



Public Member Functions

- def __init__ (self, multiprocessing.Queue inQueue, configparser.ConfigParser inConfigObject, str inURL)

Public Attributes

- good

7.1.1 Detailed Description

```
This class contains the logic necessary to parse BBC News articles
```

Definition at line 15 of file BBCWebsite.py.

7.1.2 Constructor & Destructor Documentation

7.1.2.1 __init__()

```
def websites.BBCWebsite.BBCWebsite.__init__ (
             self,
            multiprocessing.Queue inQueue,
            configparser.ConfigParser inConfigObject,
            str inURL )
```

```
Instance constructor
:param inQueue: input queue to return results
:param inConfigObject: Configuration object to read from
```

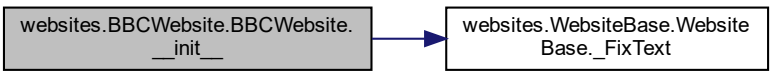
Reimplemented from websites.WebsiteBase.WebsiteBase.

Definition at line 21 of file BBCWebsite.py.

```
21     def __init__(self, inQueue: multiprocessing.Queue, inConfigObject: configparser.ConfigParser, inURL:
       str):
22         """
23         Instance constructor
24         :param inQueue: input queue to return results
25         :param inConfigObject: Configuration object to read from
26         """
27
28         # Just pass the input parameters to the base class
29         super().__init__(inQueue, inConfigObject, inURL)
30
```

References websites.WebsiteBase.WebsiteBase._FixText().

Here is the call graph for this function:



7.1.3 Member Data Documentation

7.1.3.1 good

```
websites.BBCWebsite.BBCWebsite.good
```

Definition at line 77 of file BBCWebsite.py.

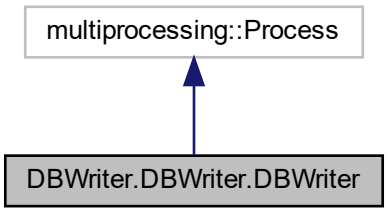
Referenced by websites.WebsiteBase.WebsiteBase.__del__().

The documentation for this class was generated from the following file:

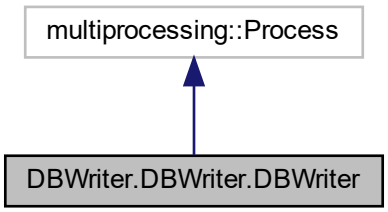
- websites/BBCWebsite.py

7.2 DBWriter.DBWriter.DBWriter Class Reference

Inheritance diagram for DBWriter.DBWriter.DBWriter:



Collaboration diagram for DBWriter.DBWriter.DBWriter:



Public Member Functions

- def __init__ (self, multiprocessing.Queue inQueue, configparser.ConfigParser inConfigObject)
- def __del__ (self)
- def WriteEntries (self, ExtractedArticle inArticleObject)
- def run (self)

Public Attributes

- good

7.2.1 Detailed Description

```
This class is responsible for checking the global queue for objects.  If any have been placed, it will then wr
them to the database.  It runs in it's own process.
```

Definition at line 16 of file DBWriter.py.

7.2.2 Constructor & Destructor Documentation

7.2.2.1 __init__()

```
def DBWriter.DBWriter.DBWriter.__init__ (
              self,
             multiprocessing.Queue inQueue,
             configparser.ConfigParser inConfigObject )
```

Perform our specific setup
:param inQueue: multiprocessing.Queue object to read from

Definition at line 23 of file DBWriter.py.

```
23     def __init__(self, inQueue: multiprocessing.Queue, inConfigObject: configparser.ConfigParser):
24         """
25         Perform our specific setup
26         :param inQueue: multiprocessing.Queue object to read from
27         """
28 
29         try:
30             multiprocessing.Process.__init__(self, group=None)
31 
32             # Flag to check if we created OK.
33             self.good = True
34 
35             # Save the global queue
36             if not inQueue:
37                 self.good = False
38 
39                 # Don't bother running if we don't have a Queue object passed in
40                 return
41 
42             self._queue = inQueue
43 
44             # DB Parameters
45             self._DBHost = str()
46             self._DBPort = str()
47             self._DBUser = str()
48             self._DBPassword = str()
49             self._DBTable = str()
50 
51             # DB Objects
52             self._DBConnection = None
53             self._DBCursor = None
54 
55             # Read the configuration
56             if not self._ReadConfiguration(inConfigObject):
57                 self.good = False
58 
59                 # If we can't read the full configuration, then no need to continue
60                 return
61 
62             # Create the database object and cursor
63             if not self._CreateDBObjects():
64                 self.good = False
65 
66                 # If we can't create the DB connection, no need to continue
67                 return
68 
69         except Exception as e:
70             print("Exception in DBWriter::__init__: {}".format(e))
71             self.good = False
72 
```

7.2.2.2 __del__()

```
def DBWriter.DBWriter.DBWriter.__del__ (
              self )
```

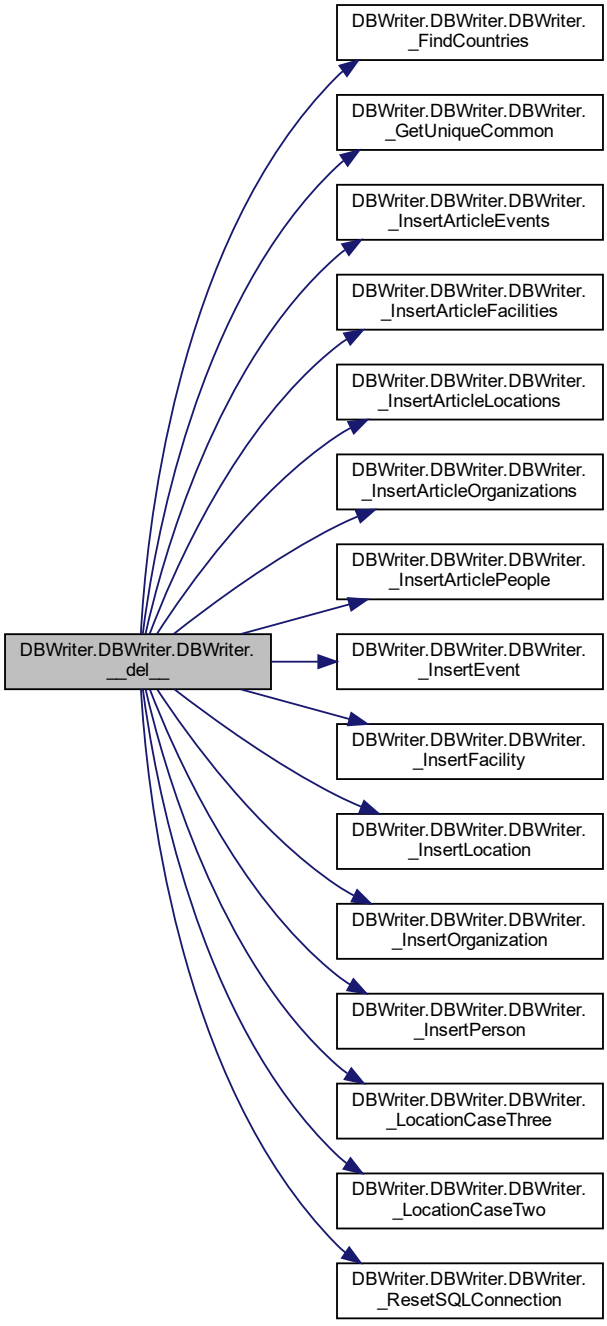
```
Clean up after ourselves
:return: None
```

Definition at line 74 of file DBWriter.py.

```
74     def __del__(self):
75         """
76         Clean up after ourselves
77         :return: None
78         """
79 
80         try:
81             # In case we have any commits outstanding.
82             if self._DBConnection:
83                 self._DBConnection.commit()
84 
85             # Now close out
86             if self._DBCursor:
87                 self._DBCursor.close()
88 
89             # No need to check here, if it's None the except will catch and ignore
90             self._DBConnection.close()
91 
92         except Exception as e:
93             print("Exception in DBWriter::__del__: {}".format(e))
94 
```

References DBWriter.DBWriter.DBWriter._DBConnection, DBWriter.DBWriter.DBWriter._DBCursor, DBWriter.↩DBWriter.DBWriter._DBHost, DBWriter.DBWriter.DBWriter._DBPassword, DBWriter.DBWriter.DBWriter._DB↩Port, DBWriter.DBWriter.DBWriter._DBTable, DBWriter.DBWriter.DBWriter._DBUser, DBWriter.DBWriter.DB↩Writer._FindCountries(), DBWriter.DBWriter.DBWriter._GetUniqueCommon(), DBWriter.DBWriter.DBWriter._↩InsertArticleEvents(), DBWriter.DBWriter.DBWriter._InsertArticleFacilities(), DBWriter.DBWriter.DBWriter._Insert↩ArticleLocations(), DBWriter.DBWriter.DBWriter._InsertArticleOrganizations(), DBWriter.DBWriter.DBWriter._↩InsertArticlePeople(), DBWriter.DBWriter.DBWriter._InsertEvent(), DBWriter.DBWriter.DBWriter._InsertFacility(), DBWriter.DBWriter.DBWriter._InsertLocation(), DBWriter.DBWriter.DBWriter._InsertOrganization(), DBWriter.D↩BWriter.DBWriter._InsertPerson(), DBWriter.DBWriter.DBWriter._LocationCaseThree(), DBWriter.DBWriter.DB↩Writer._LocationCaseTwo(), DBWriter.DBWriter.DBWriter._ResetSQLConnection(), and DBWriter.DBWriter.DB↩Writer.good.

Here is the call graph for this function:



7.2.3 Member Function Documentation

7.2.3.1 run()

```
def DBWriter.DBWriter.DBWriter.run (
              self )
```

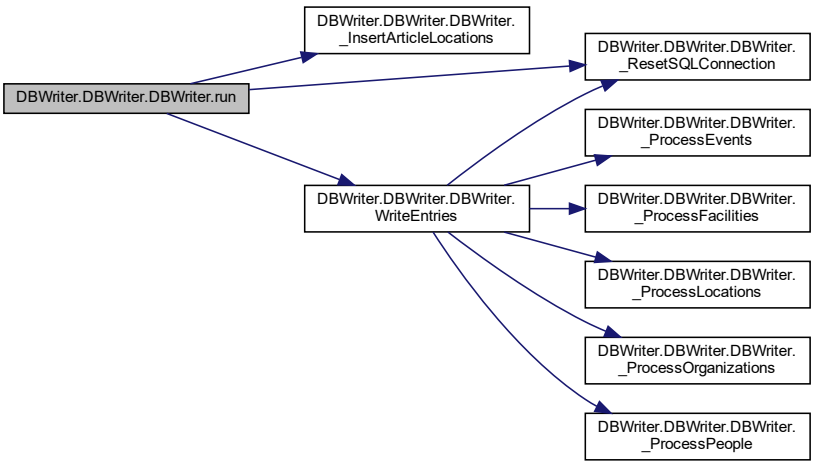
```
Override to handle the main loop of the process
:return:
```

Definition at line 653 of file DBWriter.py.

```
653     def run(self):
654         """
655         Override to handle the main loop of the process
656         :return:
657         """
658 
659         continueFlag = True
660 
661         while continueFlag:
662             articleObject = self._queue.get()
663 
664             # Check for our poison pill
665             if articleObject.articleText == "EXITCALLED":
666                 continueFlag = False
667                 continue
668 
669             self.WriteEntries(articleObject)
670 
```

References DBWriter.DBWriter.DBWriter._DBConnection, DBWriter.DBWriter.DBWriter._DBCursor, DBWriter.D↩BWriter.DBWriter._InsertArticleLocations(), DBWriter.DBWriter.DBWriter._queue, DBWriter.DBWriter.DBWriter._↩ResetSQLConnection(), and DBWriter.DBWriter.DBWriter.WriteEntries().

Here is the call graph for this function:



7.2.3.2 WriteEntries()

```
def DBWriter.DBWriter.DBWriter.WriteEntries (
              self,
             ExtractedArticle inArticleObject )
```

This is the main part of the thread that pulls from the global queue and then writes it into the database
:return:

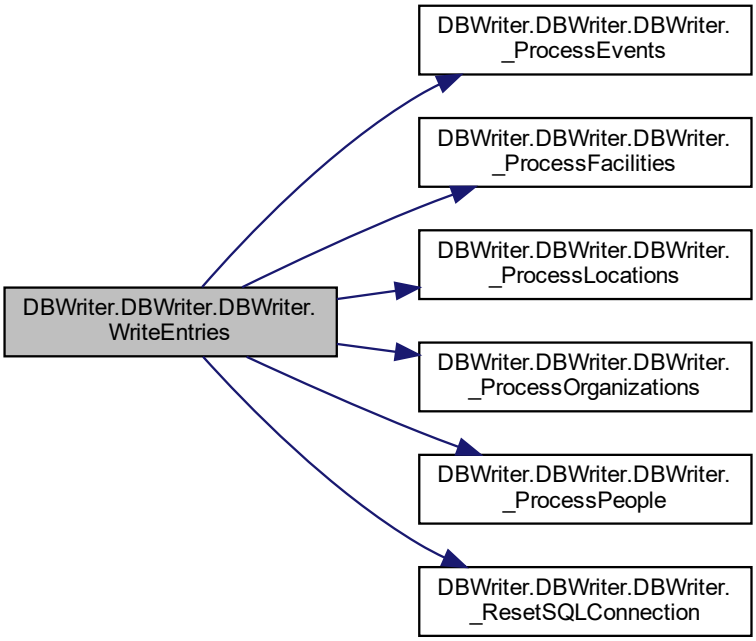
Definition at line 608 of file DBWriter.py.

```
608     def WriteEntries(self, inArticleObject: ExtractedArticle):
609         """
610         This is the main part of the thread that pulls from the global queue and then writes it into the
    database
611         :return:
612         """
613 
614         try:
615             # Push the article metadata to the DB and get the assigned identifier for it.
616             print("""
617                  INSERT INTO news_articles (article_title, article_url, article_text, website)
618                  VALUES ({}, {}, {}, {}) RETURNING id;
619                  """.format(inArticleObject.articleTitle, inArticleObject.articleURL,
    inArticleObject.articleText,
620                             inArticleObject.website))
621             self._DBCursor.execute("""
622                                    INSERT INTO news_articles (article_title, article_url, article_text,
    website)
623                                    VALUES (%s, %s, %s, %s) RETURNING id;
624                                    """,
625                                   (inArticleObject.articleTitle, inArticleObject.articleURL,
626                                    inArticleObject.articleText, inArticleObject.website))
627             self._DBConnection.commit()
628 
629             # Get the ID of the inserted object
630             articleID = self._DBCursor.fetchone()[0]
631 
632             # Now populate the article_people table
633             self._ProcessPeople(articleID, inArticleObject.people)
634 
635             # Now populate the article_facilities table
636             self._ProcessFacilities(articleID, inArticleObject.facilities)
637 
638             # Now populate the article_organizations table
639             self._ProcessOrganizations(articleID, inArticleObject.organizations)
640 
641             # Now populate the article_events table
642             self._ProcessEvents(articleID, inArticleObject.events)
643 
644             # Now populate the article_locations table
645             self._ProcessLocations(articleID, inArticleObject.locations)
646 
647         except Exception as e:
648             print("Exception in DBWriter::WriteEntries: {}".format(e))
649             self.good = False
650             self._ResetSQLConnection()
651 
```

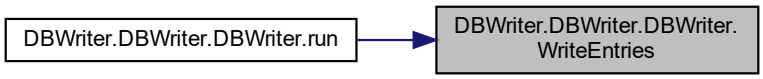
References DBWriter.DBWriter.DBWriter._DBConnection, DBWriter.DBWriter.DBWriter._DBCursor, DBWriter.↩
DBWriter.DBWriter._ProcessEvents(), DBWriter.DBWriter.DBWriter._ProcessFacilities(), DBWriter.DBWriter.↩
DBWriter._ProcessLocations(), DBWriter.DBWriter.DBWriter._ProcessOrganizations(), DBWriter.DBWriter.DB↩
Writer._ProcessPeople(), DBWriter.DBWriter.DBWriter._ResetSQLConnection(), and DBWriter.DBWriter.DB↩
Writer.good.

Referenced by DBWriter.DBWriter.DBWriter.run().

Here is the call graph for this function:



Here is the caller graph for this function:



7.2.4 Member Data Documentation

7.2.4.1 good

`DBWriter.DBWriter.DBWriter.good`

Definition at line 33 of file DBWriter.py.

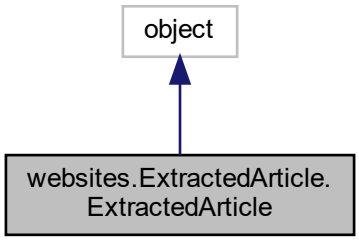
Referenced by DBWriter.DBWriter.DBWriter.__del__(), websites.WebsiteBase.WebsiteBase.__del__(), and DB↩ Writer.DBWriter.DBWriter.WriteEntries().

The documentation for this class was generated from the following file:

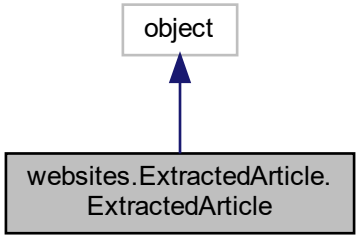
- DBWriter/DBWriter.py

7.3 websites.ExtractedArticle.ExtractedArticle Class Reference

Inheritance diagram for websites.ExtractedArticle.ExtractedArticle:



Collaboration diagram for websites.ExtractedArticle.ExtractedArticle:



Public Member Functions

- def __init__ (self)

Public Attributes

- articleTitle
- articleText
- articleURL
- website
- locations
- facilities
- people
- organizations
- events

7.3.1 Detailed Description

```
This class holds all of the information that was pulled from the article.  It is
passed to the DB thread to extract and push to the database.
```

Definition at line 9 of file ExtractedArticle.py.

7.3.2 Constructor & Destructor Documentation

7.3.2.1 __init__()

```
def websites.ExtractedArticle.ExtractedArticle.__init__ (
              self )
```

Definition at line 16 of file ExtractedArticle.py.

```
16     def __init__(self):
17         # Basic information
18         self.articleTitle = str()  # Holds the title of the article.
19         self.articleText = str()  # Holds the cleaned up full text of the article.
20         self.articleURL = str()  # URL for the article.
21         self.website = str()  # Website that the article is under (i.e., www.bbc.com).
22 
23         # Extracted information
24         self.locations = list()  # List of locations pulled from the article.
25         self.facilities = list()  # List of facilities extracted from the text.
26         self.people = list()  # List of people extracted from the article.
27         self.organizations = list()  # List of organizations extracted from the article.
28         self.events = list()  # List of events extracted from the article.
```

7.3.3 Member Data Documentation

7.3.3.1 articleText

```
websites.ExtractedArticle.ExtractedArticle.articleText
```

Definition at line 19 of file ExtractedArticle.py.

7.3.3.2 articleTitle

```
websites.ExtractedArticle.ExtractedArticle.articleTitle
```

Definition at line 18 of file ExtractedArticle.py.

7.3.3.3 articleURL

`websites.ExtractedArticle.ExtractedArticle.articleURL`

Definition at line 20 of file ExtractedArticle.py.

7.3.3.4 events

`websites.ExtractedArticle.ExtractedArticle.events`

Definition at line 28 of file ExtractedArticle.py.

7.3.3.5 facilities

`websites.ExtractedArticle.ExtractedArticle.facilities`

Definition at line 25 of file ExtractedArticle.py.

7.3.3.6 locations

`websites.ExtractedArticle.ExtractedArticle.locations`

Definition at line 24 of file ExtractedArticle.py.

7.3.3.7 organizations

`websites.ExtractedArticle.ExtractedArticle.organizations`

Definition at line 27 of file ExtractedArticle.py.

7.3.3.8 people

`websites.ExtractedArticle.ExtractedArticle.people`

Definition at line 26 of file ExtractedArticle.py.

7.3.3.9 website

`websites.ExtractedArticle.ExtractedArticle.website`

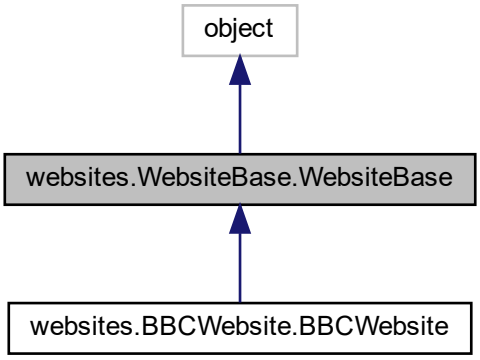
Definition at line 21 of file ExtractedArticle.py.

The documentation for this class was generated from the following file:

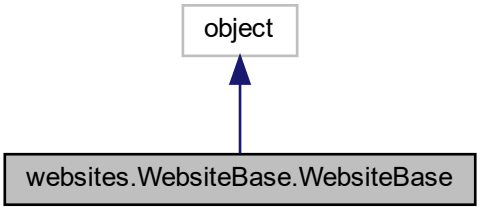
- websites/ExtractedArticle.py

7.4 websites.WebsiteBase.WebsiteBase Class Reference

Inheritance diagram for websites.WebsiteBase.WebsiteBase:



Collaboration diagram for websites.WebsiteBase.WebsiteBase:



Public Member Functions

- def __init__ (self, multiprocessing.Queue inQueue, configparser.ConfigParser inConfigObject, str inURL)
- def __del__ (self)
- def ProcessFeed (self)

Public Attributes

- good
- cursor

7.4.1 Detailed Description

```
WebsiteBase class.
This class is the base class for all of the website processing classes.  It defines their interface and provic
some common functionality for all classes.
```

Definition at line 21 of file WebsiteBase.py.

7.4.2 Constructor & Destructor Documentation

7.4.2.1 __init__()

```
def websites.WebsiteBase.WebsiteBase.__init__ (
              self,
             multiprocessing.Queue inQueue,
             configparser.ConfigParser inConfigObject,
             str inURL )
```

```
Initializer

Initializes the object.

:param inQueue: Input multiprocessing Queue object
:param inConfigObject: Input global config object
:param inURL: Input URL to scrape
```

Reimplemented in websites.BBCWebsite.BBCWebsite.

Definition at line 28 of file WebsiteBase.py.

```
28     def __init__(self, inQueue: multiprocessing.Queue, inConfigObject: configparser.ConfigParser, inURL:
       str):
29         """Initializer
30 
31         Initializes the object.
32 
33         :param inQueue: Input multiprocessing Queue object
34         :param inConfigObject: Input global config object
35         :param inURL: Input URL to scrape
36         """
37 
38         super().__init__()
39 
40         # Flag to check if we created OK.
```

```
41          self.good = True
42
43          # Spacy labels we're interested in
44          self._goodLabels = ["PERSON", "FACILITY", "FAC", "ORG", "GPE", "EVENT"]
45
46          # Save the global queue
47          if not inQueue:
48              self.good = False
49
50              # Don't bother running if we don't have a Queue object passed in
51              return
52
53          self._NLP = spacy.load("en_core_web_lg")
54
55          self._queue = inQueue
56
57          # DB Parameters
58          self._DBHost = str()
59          self._DBPort = str()
60          self._DBUser = str()
61          self._DBPassword = str()
62          self._DBTable = str()
63
64          # DB Objects
65          self._DBConnection = None
66          self._DBCursor = None
67
68          # Read the configuration
69          if not self._ReadConfiguration(inConfigObject):
70              self.good = False
71
72              # If we can't read the full configuration, then no need to continue
73              return
74
75          # Create the database object and cursor
76          if not self._CreateDBObjects():
77              self.good = False
78
79              # If we can't create the DB connection, no need to continue
80              return
81
82          # Our URL to be set in children
83          self._url = inURL
84
85          # Feed items parsed from the RSS link
86          self._feedItems = list()
87
88          # Problem entities we need to manually search for
89          self._problemEntities = self._GetProblemEntities()
90
```

7.4.2.2 __del__()

```
def websites.WebsiteBase.WebsiteBase.__del__ (
                self )
```

Destructor.

Clean up after ourselves
:return: None

Definition at line 92 of file WebsiteBase.py.

```
92      def __del__(self):
93          """Destructor.
94
95          Clean up after ourselves
96          :return: None
97          """
98
99          try:
100             # In case we have any commits outstanding.
101             if self._DBConnection:
102                 self._DBConnection.commit()
103
104             # Now close out
```

```
105             if self._DBCursor:
106                 self._DBCursor.close()
107 
108             # No need to check here, if it's None the except will catch and ignore
109             self._DBConnection.close()
110         except Exception as e:
111             print("Exception in destructor: {}".format(e))
112 
```

References DBWriter.DBWriter.DBWriter._DBConnection, websites.WebsiteBase.WebsiteBase._DBConnection, DBWriter.DBWriter.DBWriter._DBCursor, websites.WebsiteBase.WebsiteBase._DBCursor, DBWriter.DBWriter.↩ DBWriter._DBHost, websites.WebsiteBase.WebsiteBase._DBHost, DBWriter.DBWriter.DBWriter._DBPassword, websites.WebsiteBase.WebsiteBase._DBPassword, DBWriter.DBWriter.DBWriter._DBPort, websites.Website↩ Base.WebsiteBase._DBPort, DBWriter.DBWriter.DBWriter._DBTable, websites.WebsiteBase.WebsiteBase._DB↩ Table, DBWriter.DBWriter.DBWriter._DBUser, websites.WebsiteBase.WebsiteBase._DBUser, websites.Website↩ Base.WebsiteBase._fuzzy_ratio, websites.WebsiteBase.WebsiteBase.cursor, DBWriter.DBWriter.DBWriter.good, websites.WebsiteBase.WebsiteBase.good, and websites.BBCWebsite.BBCWebsite.good.

7.4.3 Member Function Documentation

7.4.3.1 ProcessFeed()

```
def websites.WebsiteBase.WebsiteBase.ProcessFeed (
                self )
```

Process the RSS feed.

Grab all of the articles from the RSS feed and process them.
:return:

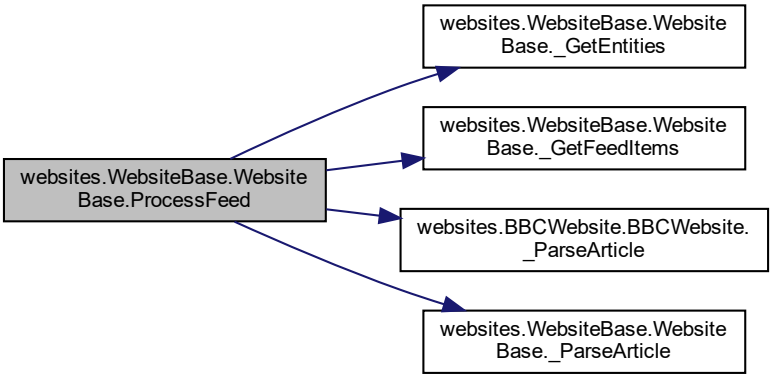
Definition at line 361 of file WebsiteBase.py.

```
361     def ProcessFeed(self):
362         """Process the RSS feed.
363 
364         Grab all of the articles from the RSS feed and process them.
365         :return:
366         """
367 
368         try:
369             extractedArticles = self._GetFeedItems()
370 
371             # Don't do anything if we have no feeds to process
372             if not extractedArticles:
373                 return
374 
375             for article in extractedArticles:
376                 article.articleText = self._ParseArticle(article.articleURL)
377 
378                 # If for whatever reason we got no text, continue on
379                 if not article.articleText:
380                     continue
381 
382                 # Have spacy do the NLP
383                 nlpEntities = self._GetEntities(article.articleText)
384 
385                 # Now populate the article
386                 article.locations.extend(nlpEntities["GPE"])
387                 article.facilities.extend(nlpEntities["FACILITY"])
388                 article.facilities.extend(nlpEntities["FAC"])
389                 article.people.extend(nlpEntities["PERSON"])
390                 article.organizations.extend(nlpEntities["ORG"])
391                 article.events.extend(nlpEntities["EVENT"])
392 
393                 # Now push into the queue
394                 self._queue.put(article)
395 
```

```
396         except Exception as e:
397             print("Got exception: {}".format(e))
398
```

References DBWriter.DBWriter.DBWriter._DBConnection, websites.WebsiteBase.WebsiteBase._DBConnection, DBWriter.DBWriter.DBWriter._DBCursor, websites.WebsiteBase.WebsiteBase._DBCursor, websites.WebsiteBase.WebsiteBase._GetEntities(), websites.WebsiteBase.WebsiteBase._GetFeedItems(), websites.BBCWebsite.BBCWebsite._ParseArticle(), websites.WebsiteBase.WebsiteBase._ParseArticle(), websites.WebsiteBase.WebsiteBase._problemEntities, DBWriter.DBWriter.DBWriter._queue, and websites.WebsiteBase.WebsiteBase._queue.

Here is the call graph for this function:



7.4.4 Member Data Documentation

7.4.4.1 cursor

`websites.WebsiteBase.WebsiteBase.cursor`

Definition at line 192 of file WebsiteBase.py.

Referenced by websites.WebsiteBase.WebsiteBase.__del__().

7.4.4.2 good

`websites.WebsiteBase.WebsiteBase.good`

Definition at line 41 of file WebsiteBase.py.

Referenced by websites.WebsiteBase.WebsiteBase.__del__().

The documentation for this class was generated from the following file:

- websites/WebsiteBase.py

Chapter 8

File Documentation

8.1 DBWriter/__init__.py File Reference

Namespaces

- DBWriter

8.2 websites/__init__.py File Reference

Namespaces

- websites

Functions

- WebsiteBase websites.WebsiteFactory (tuple inURL, multiprocessing.Queue inQueue, configparser.Config↩ Parser inConfigObject)

8.3 DBWriter/DBWriter.py File Reference

Classes

- class DBWriter.DBWriter.DBWriter

Namespaces

- DBWriter.DBWriter

8.4 newsextractor.py File Reference

Namespaces

- newsextractor

Functions

- List newsextractor.GetSubscriptions (ConfigParser inConfigObject)
- def newsextractor.AddPoisonPill ()

Variables

- newsextractor.localtime = time.asctime(time.localtime(time.time()))
- newsextractor.articleQueue = multiprocessing.Queue()
- newsextractor.plugin_path = os.path.dirname(os.path.realpath(__file__))
- newsextractor.configObject = ConfigParser()
- List newsextractor.urlList = GetSubscriptions(configObject)
- newsextractor.dbWriter = DBWriter(articleQueue, configObject)
- newsextractor.websiteObject = websites.WebsiteFactory(url, articleQueue, configObject)

8.5 README.md File Reference

8.6 requirements.txt File Reference

8.7 websites/BBCWebsite.py File Reference

Classes

- class websites.BBCWebsite.BBCWebsite

Namespaces

- websites.BBCWebsite

8.8 websites/ExtractedArticle.py File Reference

Classes

- class websites.ExtractedArticle.ExtractedArticle

Namespaces

- websites.ExtractedArticle

8.9 websites/WebsiteBase.py File Reference

Classes

- class websites.WebsiteBase.WebsiteBase

Namespaces

- websites.WebsiteBase

Index