

Design & Implementierung eines Echtzeit-Q&A-Systems als Erweiterung des IAmA-Subreddits

(Python) – Dokumentation von Crawler / Analyzer - Scripts

Benedikt Hierl
Version 1.0
Sonntag, den 25.09.2016

Table of Contents

Namespace Index	2
Class Index	3
File Index	4
a__everything_Big_CSV_analyzer	5
a_author_Information	27
a_iAMA_Commenttime_arr	29
a_iAMA_Commenttime_med	35
a_question_Answered_Yes_No_Extrema	41
a_question_Answered_Yes_No_Tier_Percentage	47
a_question_Tier_Distribution	53
a_thread_Lifespan_N_Average_Commenttime	58
a_thread_Lifespan_N_Average_Questiontime	63
ar_analyzer	68
c_crawl_Author_Information	74
c_crawl_Differences	78
c_crawl_Random_Author_Information	82
c_crawl_Threads_N_Comments	86
d_create_Big_CSV_ar	90
d_create_Big_CSV_med	96
PlotlyBarChart	102
PlotlyBarChart_5_Bars	103
Class Documentation	104
PlotlyBarChart.PlotlyBarChart	104
PlotlyBarChart_5_Bars.PlotlyBarChart5Bars	108
File Documentation	115
a__everything_Big_CSV_analyzer.py	115
a_author_Information.py	120
a_iAMA_Commenttime_arr.py	121
a_iAMA_Commenttime_med.py	122
a_question_Answered_Yes_No_Extrema.py	123
a_question_Answered_Yes_No_Tier_Percentage.py	124
a_question_Tier_Distribution.py	125
a_thread_Lifespan_N_Average_Commenttime.py	126
a_thread_Lifespan_N_Average_Questiontime.py	127
ar_analyzer.py	128
c_crawl_Author_Information.py	130
c_crawl_Differences.py	131
c_crawl_Random_Author_Information.py	132
c_crawl_Threads_N_Comments.py	133
d_create_Big_CSV_ar.py	134
d_create_Big_CSV_med.py	135
PlotlyBarChart.py	136
PlotlyBarChart_5_Bars.py	137
Index	138

Namespace Index

Packages

Here are the packages with brief descriptions (if available):

<u>a everything Big CSV analyzer</u>	5
<u>a author Information</u>	27
<u>a iAMA Commenttime arr</u>	29
<u>a iAMA Commenttime med</u>	35
<u>a question Answered Yes No Extrema</u>	41
<u>a question Answered Yes No Tier Percentage</u>	47
<u>a question Tier Distribution</u>	53
<u>a thread Lifespan N Average Commenttime</u>	58
<u>a thread Lifespan N Average Questiontime</u>	63
<u>ar analyzer</u>	68
<u>c crawl Author Information</u>	74
<u>c crawl Differences</u>	78
<u>c crawl Random Author Information</u>	82
<u>c crawl Threads N Comments</u>	86
<u>d create Big CSV ar</u>	90
<u>d create Big CSV med</u>	96
<u>PlotlyBarChart</u>	102
<u>PlotlyBarChart 5 Bars</u>	103

Class Index

Class List

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

<u>PlotlyBarChart.PlotlyBarChart</u>	104
<u>PlotlyBarChart_5_Bars.PlotlyBarChart5Bars</u>	108

File Index

File List

Here is a list of all files with brief descriptions:

<u>a everything Big CSV analyzer.py</u>	115
<u>a author Information.py</u>	120
<u>a iAMA Commenttime arr.py</u>	121
<u>a iAMA Commenttime med.py</u>	122
<u>a question Answered Yes No Extrema.py</u>	123
<u>a question Answered Yes No Tier Percentage.py</u>	124
<u>a question Tier Distribution.py</u>	125
<u>a thread Lifespan N Average Commenttime.py</u>	126
<u>a thread Lifespan N Average Questiontime.py</u>	127
<u>ar analyzer.py</u>	128
<u>c crawl Author Information.py</u>	130
<u>c crawl Differences.py</u>	131
<u>c crawl Random Author Information.py</u>	132
<u>c crawl Threads N Comments.py</u>	133
<u>d create Big CSV ar.py</u>	134
<u>d create Big CSV med.py</u>	135
<u>PlotlyBarChart.py</u>	136
<u>PlotlyBarChart 5 Bars.py</u>	137

Namespace Documentation

a__everything_Big_CSV_analyzer Namespace Reference

Functions

- def [relation_question_upvotes_with_amount_of_questions_answered_by_iam_a_host\(\)](#)
- def [average_means_of_values_f_threads\(\)](#)
- def [relation_thread_upvotes_with_amount_of_comments\(\)](#)
- def [relation_thread_upvotes_with_amount_of_questions\(\)](#)
- def [relation_thread_downvotes_with_amount_of_comments\(\)](#)
- def [relation_thread_downvotes_with_amount_of_questions\(\)](#)
- def [relation_thread_upvotes_and_iam_a_host_response_time_comments\(\)](#)
- def [relation_thread_upvotes_and_iam_a_host_response_time_questions\(\)](#)
- def [relation_thread_downvotes_and_iam_a_host_response_time_comments\(\)](#)
- def [relation_thread_downvotes_and_iam_a_host_response_time_questions\(\)](#)
- def [relation_thread_lifespan_to_last_comment_and_amount_of_comments\(\)](#)
- def [relation_thread_lifespan_to_last_comment_and_amount_of_questions\(\)](#)
- def [relation_thread_lifespan_to_last_question_and_amount_of_comments\(\)](#)
- def [relation_thread_lifespan_to_last_question_and_amount_of_question\(\)](#)
- def [relation_thread_lifespan_to_last_comment_and_iam_a_host_response_time_to_comments\(\)](#)
- def [relation_thread_lifespan_to_last_comment_and_iam_a_host_response_time_to_questions\(\)](#)
- def [relation_thread_lifespan_to_last_question_and_iam_a_host_response_time_to_comments\(\)](#)
- def [relation_thread_lifespan_to_last_question_and_iam_a_host_response_time_to_questions\(\)](#)
- def [relation_thread_reaction_time_comments_and_iam_a_host_response_time_to_comments\(\)](#)
- def [relation_thread_reaction_time_comments_and_iam_a_host_response_time_to_questions\(\)](#)
- def [relation_thread_reaction_time_questions_and_iam_a_host_response_time_to_comments\(\)](#)
- def [relation_thread_reaction_time_questions_and_iam_a_host_response_time_to_questions\(\)](#)
- def [relation_thread_reaction_time_comments_and_amount_of_comments_the_iam_a_host_answered_to\(\)](#)
- def [relation_thread_reaction_time_comments_and_amount_of_questions_the_iam_a_host_answered_to\(\)](#)
- def [relation_thread_reaction_time_questions_and_amount_of_comments_the_iam_a_host_answered_to\(\)](#)
- def [relation_thread_reaction_time_questions_and_amount_of_questions_the_iam_a_host_answered_to\(\)](#)
- def [relation_thread_amount_of_questioners_total_and_num_questions_answered_by_iam_a_host\(\)](#)
- def [realation_thread_amount_of_commentators_total_and_num_comments_answered_by_iam_a_host\(\)](#)
- def [relation_thread_amount_of_questions_and_amount_questions_answered_by_iam_a_host\(\)](#)
- def [thread_overall_correlation\(\)](#)
- def [question_overall_correlation\(\)](#)
- def [average_means_of_values_f_authors\(\)](#)
- def [median_of_values_f_authors_randomized\(\)](#)
- def [median_of_values_f_authors\(\)](#)
- def [arr_of_values_f_authors\(\)](#)
- def [calculate_t_tests_of_author_values\(\)](#)

Variables

- [author_information_iam_a](#)
- [author_information_iam_a_sampleset_randomized](#)
- [author_information_random](#)
- [author_information_random_sampleset_randomized](#)
- [thread_information](#)
- [question_information](#)
- [inplace](#)
- [author_amount_creation_iam_a_threads](#) = [author_information_iam_a](#)['amount_creation_iam_a_threads']

- [author amount creation other threads](#) = [author information iama](#)['amount_creation_other_threads']
- [author amount of comments except iama](#) = [author information iama](#)['amount_of_comments_except_iama']
- [author amount of comments iama](#) = [author information iama](#)['amount_of_comments_iama']
- [author author birth date](#) = [author information iama](#)['author_birth_date']
- [author author comment karma amount](#) = [author information iama](#)['author_comment_karma_amount']
- [author author link karma amount](#) = [author information iama](#)['author_link_karma_amount']
- [author author name](#) = [author information iama](#)['author_name']
- [author comment creation every x sec](#) = [author information iama](#)['comment_creation_every_x_sec']
- [author thread creation every x sec](#) = [author information iama](#)['thread_creation_every_x_sec']
- [author time acc birth first iama thread](#) = [author information iama](#)['time_acc_birth_first_iama_thread']
- [author time diff acc creation n first comment](#) = [author information iama](#)['time_diff_acc_creation_n_first_comment']
- [author time diff acc creation n first thread](#) = [author information iama](#)['time_diff_acc_creation_n_first_thread']
- [author amount creation iama threads randomized](#) = [author information iama sampleset randomized](#)['amount_creation_iama_threads']
- [author amount creation other threads randomized](#) = [author information iama sampleset randomized](#)['amount_creation_other_threads']
- [author amount of comments except iama randomized](#) = [author information iama sampleset randomized](#)['amount_of_comments_except_iama']
- [author amount of comments iama randomized](#) = [author information iama sampleset randomized](#)['amount_of_comments_iama']
- [author author birth date randomized](#) = [author information iama sampleset randomized](#)['author_birth_date']
- [author author comment karma amount randomized](#) = [author information iama sampleset randomized](#)['author_comment_karma_amount']
- [author author link karma amount randomized](#) = [author information iama sampleset randomized](#)['author_link_karma_amount']
- [author author name randomized](#) = [author information iama sampleset randomized](#)['author_name']
- [author comment creation every x sec randomized](#) = [author information iama sampleset randomized](#)['comment_creation_every_x_sec']
- [author thread creation every x sec randomized](#) = [author information iama sampleset randomized](#)['thread_creation_every_x_sec']
- [author time acc birth first iama thread randomized](#) = [author information iama sampleset randomized](#)['time_acc_birth_first_iama_thread']
- [author time diff acc creation n first comment randomized](#) = [author information iama sampleset randomized](#)['time_diff_acc_creation_n_first_comment']
- [author time diff acc creation n first thread randomized](#) = [author information iama sampleset randomized](#)['time_diff_acc_creation_n_first_thread']
- [random author amount creation iama threads](#) = [author information random](#)['amount_creation_iama_threads']
- [random author amount creation other threads](#) = [author information random](#)['amount_creation_other_threads']
- [random author amount of comments except iama](#) = [author information random](#)['amount_of_comments_except_iama']
- [random author amount of comments iama](#) = [author information random](#)['amount_of_comments_iama']
- [random author author birth date](#) = [author information random](#)['author_birth_date']
- [random author author comment karma amount](#) = [author information random](#)['author_comment_karma_amount']
- [random author author link karma amount](#) = [author information random](#)['author_link_karma_amount']
- [random author author name](#) = [author information random](#)['author_name']
- [random author comment creation every x sec](#) = [author information random](#)['comment_creation_every_x_sec']
- [random author thread creation every x sec](#) = [author information random](#)['thread_creation_every_x_sec']
- [random author time acc birth first iama thread](#) = [author information random](#)['time_acc_birth_first_iama_thread']
- [random author time diff acc creation n first comment](#) = \

- [random author time diff acc creation n first thread](#)
- [random author amount creation iama threads randomized](#) = [author information random sampleset randomized](#)['amount_creation_iama_threads']
- [random author amount creation other threads randomized](#) = [author information random sampleset randomized](#)['amount_creation_other_threads']
- [random author amount of comments except iama randomized](#) = [author information random sampleset randomized](#)['amount_of_comments_except_iama']
- [random author amount of comments iama randomized](#) = [author information random sampleset randomized](#)['amount_of_comments_iama']
- [random author author birth date randomized](#) = [author information random sampleset randomized](#)['author_birth_date']
- [random author author comment karma amount randomized](#) = [author information random sampleset randomized](#)['author_comment_karma_amount']
- [random author author link karma amount randomized](#) = [author information random sampleset randomized](#)['author_link_karma_amount']
- [random author author name randomized](#) = [author information random sampleset randomized](#)['author_name']
- [random author comment creation every x sec randomized](#) = [author information random sampleset randomized](#)['comment_creation_every_x_sec']
- [random author thread creation every x sec randomized](#) = [author information random sampleset randomized](#)['thread_creation_every_x_sec']
- [random author time acc birth first iama thread randomized](#) = [author information random sampleset randomized](#)['time_acc_birth_first_iama_thread']
- [random author time diff acc creation n first comment randomized](#) = \
- [random author time diff acc creation n first thread randomized](#)
- [question ups](#) = [question information](#)['Question ups']
- [question answered by iAMA host](#) = [question information](#)['Question answered by iAMA host']
- [thread year](#) = [thread information](#)['Year']
- [thread id](#) = [thread information](#)['Thread id']
- [thread author](#) = [thread information](#)['Thread author']
- [thread ups](#) = [thread information](#)['Thread ups']
- [thread downs](#) = [thread information](#)['Thread downs']
- [thread creation time stamp](#) = [thread information](#)['Thread creation time stamp']
- [thread average comment vote score total](#)
- [thread average comment vote score tier 1](#)
- [thread average comment vote score tier x](#)
- [thread average question vote score total](#)
- [thread average question vote score tier 1](#)
- [thread average question vote score tier x](#)
- [thread num comments total skewed](#)
- [thread num comments total](#) = [thread information](#)['Thread num comments total']
- [thread num comments tier 1](#) = [thread information](#)['Thread num comments tier 1']
- [thread num comments tier x](#) = [thread information](#)['Thread num comments tier x']
- [thread num questions total](#) = [thread information](#)['Thread num questions total']
- [thread num questions tier 1](#) = [thread information](#)['Thread num questions tier 1']
- [thread num questions tier x](#) = [thread information](#)['Thread num questions tier x']
- [thread num questions answered by iama host total](#)
- [thread num questions answered by iama host tier 1](#)
- [thread num questions answered by iama host tier x](#)
- [thread num comments answered by iama host total](#)
- [thread num comments answered by iama host tier 1](#)
- [thread num comments answered by iama host tier x](#)
- [thread average reaction time between comments total](#)
- [thread average reaction time between comments tier 1](#)
- [thread average reaction time between comments tier x](#)

- [thread average reaction time between questions total](#)
 - [thread average reaction time between questions tier 1](#)
 - [thread average reaction time between questions tier x](#)
 - [thread average response to comment time iama host total](#)
 - [thread average response to comment time iama host tier 1](#)
 - [thread average response to comment time iama host tier x](#)
 - [thread average response to question time iama host total](#)
 - [thread average response to question time iama host tier 1](#)
 - [thread average response to question time iama host tier x](#)
 - [thread amount of questioners total](#)
 - [thread amount of questioners tier 1](#)
 - [thread amount of questioners tier x](#)
 - [thread amount of commentators total](#)
 - [thread amount of commentators tier 1](#)
 - [thread amount of commentators tier x](#)
 - [thread life span until last comment](#)
 - [thread life span until last question](#)
-

Function Documentation

def a__everything_Big_CSV_analyzer.arr_of_values_f_authors ()

Definition at line 3671 of file a__everything_Big_CSV_analyzer.py.

def a__everything_Big_CSV_analyzer.average_means_of_values_f_authors ()

Calculation of the average means of different values for author data

Args:
_
Returns:
_

Definition at line 3252 of file a__everything_Big_CSV_analyzer.py.

def a__everything_Big_CSV_analyzer.average_means_of_values_f_threads ()

Calculation of the average means of different values

Args:
_
Returns:
_

Definition at line 339 of file a__everything_Big_CSV_analyzer.py.

def a__everything_Big_CSV_analyzer.calculate_t_tests_of_author_values ()

Definition at line 3812 of file a__everything_Big_CSV_analyzer.py.

def a__everything_Big_CSV_analyzer.median_of_values_f_authors ()

Definition at line 3529 of file a__everything_Big_CSV_analyzer.py.

def a__everything_Big_CSV_analyzer.median_of_values_f_authors_randomized ()

Definition at line 3402 of file a__everything_Big_CSV_analyzer.py.

def a__everything_Big_CSV_analyzer.question_overall_correlation ()

```
Calculation of the correlation of every column with every column for the questions
Args:
-
Returns:
-
```

Definition at line 3240 of file a__everything_Big_CSV_analyzer.py.

def
a__everything_Big_CSV_analyzer.realation_thread_amount_of_commentators_total_and_num_comments_answered_by_iama_host ()

```
Calculation of the correlation amount of commentators per thread <-> amount of questions answered by iama host
Args:
-
Returns:
-
```

Definition at line 3003 of file a__everything_Big_CSV_analyzer.py.

def
a__everything_Big_CSV_analyzer.relation_question_upvotes_with_amount_of_questions_answered_by_iama_host ()

```
Calculation of the correlation question upvotes <-> amount of questions answered by the iama host
Args:
-
Returns:
-
```

Definition at line 305 of file a__everything_Big_CSV_analyzer.py.

def
a__everything_Big_CSV_analyzer.relation_thread_amount_of_questioners_total_and_num_questions_answered_by_iama_host ()

```
Calculation of the correlation amount of questioners per thread <-> amount of questions answered by iama host
Args:
```

```
-  
Returns:  
-
```

Definition at line 2875 of file a__everything_Big_CSV_analyzer.py.

```
def  
a__everything_Big_CSV_analyzer.relation_thread_amount_of_questions_and_amount_questions_  
answered_by_iama_host ()
```

```
Calculation of the amount of questions asked <-> amount of questions answered by iama host  
Args:  
-  
Returns:  
-
```

Definition at line 3131 of file a__everything_Big_CSV_analyzer.py.

```
def  
a__everything_Big_CSV_analyzer.relation_thread_downvotes_and_iama_host_response_time_co  
mments ()
```

```
Calculation of the correlation thread downvotes <-> iama host repsonse time to comments  
Args:  
-  
Returns:  
-
```

Definition at line 886 of file a__everything_Big_CSV_analyzer.py.

```
def  
a__everything_Big_CSV_analyzer.relation_thread_downvotes_and_iama_host_response_time_qu  
estions ()
```

```
Calculation of the correlation thread downvotes <-> iama host repsonse time to questions  
Args:  
-  
Returns:  
-
```

Definition at line 972 of file a__everything_Big_CSV_analyzer.py.

```
def a__everything_Big_CSV_analyzer.relation_thread_downvotes_with_amount_of_comments ()
```

```
Calculation of the correlation thread downvotes <-> amount of comments  
Args:  
-  
Returns:  
-
```

Definition at line 584 of file a__everything_Big_CSV_analyzer.py.

def a__everything_Big_CSV_analyzer.relation_thread_downvotes_with_amount_of_questions ()

```
Calculation of the correlation thread downvotes <-> amount of questions  
Args:  
-  
Returns:  
-
```

Definition at line 651 of file a__everything_Big_CSV_analyzer.py.

**def
a__everything_Big_CSV_analyzer.relation_thread_lifespan_to_last_comment_and_amount_of_comments ()**

```
Calculation of the correlation thread life span (until last comment) <-> amount of comments  
Args:  
-  
Returns:  
-
```

Definition at line 1060 of file a__everything_Big_CSV_analyzer.py.

**def
a__everything_Big_CSV_analyzer.relation_thread_lifespan_to_last_comment_and_amount_of_questions ()**

```
Calculation of the correlation thread life span (until last comment) <-> amount of questions  
Args:  
-  
Returns:  
-
```

Definition at line 1127 of file a__everything_Big_CSV_analyzer.py.

**def
a__everything_Big_CSV_analyzer.relation_thread_lifespan_to_last_comment_and_iama_host_response_time_to_comments ()**

```
Calculation of the correlation thread life span (until last comment) <-> iama host repsonse time to comments  
Args:  
-  
Returns:  
-
```

Definition at line 1335 of file a__everything_Big_CSV_analyzer.py.

**def
a__everything_Big_CSV_analyzer.relation_thread_lifespan_to_last_comment_and_iama_host_response_time_to_questions ()**

```
Calculation of the correlation thread life span (until last comment) <-> iama host repsonse time to questions
```

```
Args:
-
Returns:
-
```

Definition at line 1463 of file a__everything_Big_CSV_analyzer.py.

```
def
a__everything_Big_CSV_analyzer.relation_thread_lifespan_to_last_question_and_amount_of_comments ()
```

```
Calculation of the correlation thread life span (until last question) <-> amount of comments
```

```
Args:
-
Returns:
-
```

Definition at line 1194 of file a__everything_Big_CSV_analyzer.py.

```
def
a__everything_Big_CSV_analyzer.relation_thread_lifespan_to_last_question_and_amount_of_question ()
```

```
Calculation of the correlation thread life span (until last question) <-> amount of question
```

```
Args:
-
Returns:
-
```

Definition at line 1261 of file a__everything_Big_CSV_analyzer.py.

```
def
a__everything_Big_CSV_analyzer.relation_thread_lifespan_to_last_question_and_iama_host_response_time_to_comments ()
```

```
Calculation of the correlation thread life span (until last question) <-> and iama host repsonse time to comments
```

```
Args:
-
Returns:
-
```

Definition at line 1591 of file a__everything_Big_CSV_analyzer.py.

```
def
a__everything_Big_CSV_analyzer.relation_thread_lifespan_to_last_question_and_iama_host_response_time_to_questions ()
```

```
Calculation of the correlation thread life span (until last question) <-> iama host repsonse time to questions
```

```
Args:
-
Returns:
-
```

Definition at line 1719 of file a__everything_Big_CSV_analyzer.py.

```
def
a__everything_Big_CSV_analyzer.relation_thread_reaction_time_comments_and_amount_of_comments_the_iama_host_answered_to ()
```

```
Calculation of the correlation thread reaction time between comments <-> amount of comments the iama host reacted to

Args:
-
Returns:
-
```

Definition at line 2359 of file a__everything_Big_CSV_analyzer.py.

```
def
a__everything_Big_CSV_analyzer.relation_thread_reaction_time_comments_and_amount_of_questions_the_iama_host_answered_to ()
```

```
Calculation of the correlation thread reaction time between comments <-> amount of questions the iama host reacted to

Args:
-
Returns:
-
```

Definition at line 2488 of file a__everything_Big_CSV_analyzer.py.

```
def
a__everything_Big_CSV_analyzer.relation_thread_reaction_time_comments_and_iama_host_response_time_to_comments ()
```

```
Calculation of the correlation thread reaction time between comments <-> iama host response time to comments

Args:
-
Returns:
-
```

Definition at line 1847 of file a__everything_Big_CSV_analyzer.py.

```
def
a__everything_Big_CSV_analyzer.relation_thread_reaction_time_comments_and_iama_host_response_time_to_questions ()
```

```
Calculation of the correlation thread reaction time between comments <-> iama host response time to questions
```



```
Args:
-
Returns:
-
```

Definition at line 1975 of file a__everything_Big_CSV_analyzer.py.

```
def
a__everything_Big_CSV_analyzer.relation_thread_reaction_time_questions_and_amount_of_comments_the_iama_host_answered_to ()
```

```
Calculation of the correlation thread reaction time between questions <-> amount of comments the iama host reacted to
```

```
Args:
-
Returns:
-
```

Definition at line 2617 of file a__everything_Big_CSV_analyzer.py.

```
def
a__everything_Big_CSV_analyzer.relation_thread_reaction_time_questions_and_amount_of_questions_the_iama_host_answered_to ()
```

```
Calculation of the correlation thread reaction time between questions <-> amount of questions the iama host reacted to
```

```
Args:
-
Returns:
-
```

Definition at line 2746 of file a__everything_Big_CSV_analyzer.py.

```
def
a__everything_Big_CSV_analyzer.relation_thread_reaction_time_questions_and_iama_host_response_time_to_comments ()
```

```
Calculation of the correlation thread reaction time between questions <-> iama host response time to comments
```

```
Args:
-
Returns:
-
```

Definition at line 2103 of file a__everything_Big_CSV_analyzer.py.

```
def
a__everything_Big_CSV_analyzer.relation_thread_reaction_time_questions_and_iama_host_response_time_to_questions ()
```

```
Calculation of the correlation thread reaction time between questions <-> iama host response time to questions
```

```
Args:
-
Returns:
-
```

Definition at line 2231 of file a__everything_Big_CSV_analyzer.py.

```
def
a__everything_Big_CSV_analyzer.relation_thread_upvotes_and_iama_host_response_time_comments ()
```

```
Calculation of the correlation thread upvotes <-> iama host repsonse time to comments

Args:
-
Returns:
-
```

Definition at line 718 of file a__everything_Big_CSV_analyzer.py.

```
def
a__everything_Big_CSV_analyzer.relation_thread_upvotes_and_iama_host_response_time_questions ()
```

```
Calculation of the correlation thread upvotes <-> iama host repsonse time to questions

Args:
-
Returns:
-
```

Definition at line 800 of file a__everything_Big_CSV_analyzer.py.

```
def a__everything_Big_CSV_analyzer.relation_thread_upvotes_with_amount_of_comments ()
```

```
Calculation of the correlation thread upvotes <-> amount of comments

Args:
-
Returns:
-
```

Definition at line 449 of file a__everything_Big_CSV_analyzer.py.

```
def a__everything_Big_CSV_analyzer.relation_thread_upvotes_with_amount_of_questions ()
```

```
Calculation of the correlation thread upvotes <-> amount of questions

Args:
-
Returns:
-
```

Definition at line 517 of file a__everything_Big_CSV_analyzer.py.

def a__everything_Big_CSV_analyzer.thread_overall_correlation ()

```
Calculation of the correlation of every column with every column for the threads  
Args:  
-  
Returns:  
-
```

Definition at line 3228 of file a__everything_Big_CSV_analyzer.py.

Variable Documentation

a__everything_Big_CSV_analyzer.author_amount_creation_iama_threads =
[author information iama](#)['amount_creation_iama_threads']

Definition at line 134 of file a__everything_Big_CSV_analyzer.py.

a__everything_Big_CSV_analyzer.author_amount_creation_iama_threads_randomized =
[author information iama sampleset randomized](#)['amount_creation_iama_threads']

Definition at line 149 of file a__everything_Big_CSV_analyzer.py.

a__everything_Big_CSV_analyzer.author_amount_creation_other_threads =
[author information iama](#)['amount_creation_other_threads']

Definition at line 135 of file a__everything_Big_CSV_analyzer.py.

a__everything_Big_CSV_analyzer.author_amount_creation_other_threads_randomized =
[author information iama sampleset randomized](#)['amount_creation_other_threads']

Definition at line 150 of file a__everything_Big_CSV_analyzer.py.

a__everything_Big_CSV_analyzer.author_amount_of_comments_except_iama =
[author information iama](#)['amount_of_comments_except_iama']

Definition at line 136 of file a__everything_Big_CSV_analyzer.py.

a__everything_Big_CSV_analyzer.author_amount_of_comments_except_iama_randomized =
[author information iama sampleset randomized](#)['amount_of_comments_except_iama']

Definition at line 151 of file a__everything_Big_CSV_analyzer.py.

a__everything_Big_CSV_analyzer.author_amount_of_comments_iama =
[author information iama](#)['amount_of_comments_iama']

Definition at line 137 of file a__everything_Big_CSV_analyzer.py.

a__everything_Big_CSV_analyzer.author_amount_of_comments_iam_a_randomized = [author_information_iam_a_sampleset_randomized](#)['amount_of_comments_iam_a']

Definition at line 152 of file a__everything_Big_CSV_analyzer.py.

a__everything_Big_CSV_analyzer.author_author_birth_date = [author_information_iam_a](#)['author_birth_date']

Definition at line 138 of file a__everything_Big_CSV_analyzer.py.

a__everything_Big_CSV_analyzer.author_author_birth_date_randomized = [author_information_iam_a_sampleset_randomized](#)['author_birth_date']

Definition at line 153 of file a__everything_Big_CSV_analyzer.py.

a__everything_Big_CSV_analyzer.author_author_comment_karma_amount = [author_information_iam_a](#)['author_comment_karma_amount']

Definition at line 139 of file a__everything_Big_CSV_analyzer.py.

a__everything_Big_CSV_analyzer.author_author_comment_karma_amount_randomized = [author_information_iam_a_sampleset_randomized](#)['author_comment_karma_amount']

Definition at line 154 of file a__everything_Big_CSV_analyzer.py.

a__everything_Big_CSV_analyzer.author_author_link_karma_amount = [author_information_iam_a](#)['author_link_karma_amount']

Definition at line 140 of file a__everything_Big_CSV_analyzer.py.

a__everything_Big_CSV_analyzer.author_author_link_karma_amount_randomized = [author_information_iam_a_sampleset_randomized](#)['author_link_karma_amount']

Definition at line 155 of file a__everything_Big_CSV_analyzer.py.

a__everything_Big_CSV_analyzer.author_author_name = [author_information_iam_a](#)['author_name']

Definition at line 141 of file a__everything_Big_CSV_analyzer.py.

a__everything_Big_CSV_analyzer.author_author_name_randomized = [author_information_iam_a_sampleset_randomized](#)['author_name']

Definition at line 156 of file a__everything_Big_CSV_analyzer.py.

a__everything_Big_CSV_analyzer.author_comment_creation_every_x_sec = [author_information_iam_a](#)['comment_creation_every_x_sec']

Definition at line 142 of file a__everything_Big_CSV_analyzer.py.

a__everything_Big_CSV_analyzer.author_comment_creation_every_x_sec_randomized = [author_information_iama_sampleset_randomized](#)['comment_creation_every_x_sec']

Definition at line 157 of file a__everything_Big_CSV_analyzer.py.

a__everything_Big_CSV_analyzer.author_information_iama

```
Initial value: 1 = pandas.read_csv(  
2     'a_author_information_iama.csv',  
3     sep=',',  
4     na_values="None")
```

Definition at line 78 of file a__everything_Big_CSV_analyzer.py.

a__everything_Big_CSV_analyzer.author_information_iama_sampleset_randomized

```
Initial value: 1 = author_information_iama.ix[np.random.choice  
2 (author_information_iama.index.values, 10000)]
```

Definition at line 84 of file a__everything_Big_CSV_analyzer.py.

a__everything_Big_CSV_analyzer.author_information_random

```
Initial value: 1 = pandas.read_csv(  
2     'a_author_Information_random.csv',  
3     sep=',',  
4     na_values="None")
```

Definition at line 90 of file a__everything_Big_CSV_analyzer.py.

a__everything_Big_CSV_analyzer.author_information_random_sampleset_randomized

```
Initial value: 1 = author_information_random.ix[np.random.choice  
2 (author_information_random.index.values, 10000)]
```

Definition at line 96 of file a__everything_Big_CSV_analyzer.py.

a__everything_Big_CSV_analyzer.author_thread_creation_every_x_sec = [author_information_iama](#)['thread_creation_every_x_sec']

Definition at line 143 of file a__everything_Big_CSV_analyzer.py.

a__everything_Big_CSV_analyzer.author_thread_creation_every_x_sec_randomized = [author_information_iama_sampleset_randomized](#)['thread_creation_every_x_sec']

Definition at line 158 of file a__everything_Big_CSV_analyzer.py.

a__everything_Big_CSV_analyzer.author_time_acc_birth_first_iama_thread = [author_information_iama](#)['time_acc_birth_first_iama_thread']

Definition at line 144 of file a__everything_Big_CSV_analyzer.py.

a__everything_Big_CSV_analyzer.author_time_acc_birth_first_iama_thread_randomized = [author_information_iama_sampleset_randomized](#)['time_acc_birth_first_iama_thread']

Definition at line 159 of file a__everything_Big_CSV_analyzer.py.

a__everything_Big_CSV_analyzer.author_time_diff_acc_creation_n_first_comment =
[author information iama](#)['time_diff_acc_creation_n_first_comment']

Definition at line 145 of file a__everything_Big_CSV_analyzer.py.

a__everything_Big_CSV_analyzer.author_time_diff_acc_creation_n_first_comment_randomized =
[author information iama sampleset randomized](#)['time_diff_acc_creation_n_first_comment']

Definition at line 160 of file a__everything_Big_CSV_analyzer.py.

a__everything_Big_CSV_analyzer.author_time_diff_acc_creation_n_first_thread =
[author information iama](#)['time_diff_acc_creation_n_first_thread']

Definition at line 146 of file a__everything_Big_CSV_analyzer.py.

a__everything_Big_CSV_analyzer.author_time_diff_acc_creation_n_first_thread_randomized =
[author information iama sampleset randomized](#)['time_diff_acc_creation_n_first_thread']

Definition at line 161 of file a__everything_Big_CSV_analyzer.py.

a__everything_Big_CSV_analyzer.inplace

Definition at line 117 of file a__everything_Big_CSV_analyzer.py.

a__everything_Big_CSV_analyzer.question_answered_by_iAMA_host =
[question information](#)['Question answered by iAMA host']

Definition at line 199 of file a__everything_Big_CSV_analyzer.py.

a__everything_Big_CSV_analyzer.question_information

```
Initial value: 1 = pandas.read_csv(  
2     'a question Answered Yes No Tier Percentage 2009 until 2016 ALL tier any.csv',  
3     sep=',',  
4     na_values="None",  
5     low_memory=False)
```

Definition at line 110 of file a__everything_Big_CSV_analyzer.py.

a__everything_Big_CSV_analyzer.question_ups = [question information](#)['Question ups']

Definition at line 198 of file a__everything_Big_CSV_analyzer.py.

a__everything_Big_CSV_analyzer.random_author_amount_creation_iama_threads =
[author information random](#)['amount_creation_iama_threads']

Definition at line 164 of file a__everything_Big_CSV_analyzer.py.

a__everything_Big_CSV_analyzer.random_author_amount_creation_iama_threads_randomized =
[author information random sampleset randomized](#)['amount_creation_iama_threads']

Definition at line 182 of file a__everything_Big_CSV_analyzer.py.

```
a__everything_Big_CSV_analyzer.random_author_amount_creation_other_threads =  
author\_information\_random['amount_creation_other_threads']
```

Definition at line 165 of file a__everything_Big_CSV_analyzer.py.

```
a__everything_Big_CSV_analyzer.random_author_amount_creation_other_threads_randomized =  
author\_information\_random\_sampleset\_randomized['amount_creation_other_threads']
```

Definition at line 183 of file a__everything_Big_CSV_analyzer.py.

```
a__everything_Big_CSV_analyzer.random_author_amount_of_comments_except_iama =  
author\_information\_random['amount_of_comments_except_iama']
```

Definition at line 166 of file a__everything_Big_CSV_analyzer.py.

```
a__everything_Big_CSV_analyzer.random_author_amount_of_comments_except_iama_randomize  
d = author\_information\_random\_sampleset\_randomized['amount_of_comments_except_iama']
```

Definition at line 184 of file a__everything_Big_CSV_analyzer.py.

```
a__everything_Big_CSV_analyzer.random_author_amount_of_comments_iama =  
author\_information\_random['amount_of_comments_iama']
```

Definition at line 167 of file a__everything_Big_CSV_analyzer.py.

```
a__everything_Big_CSV_analyzer.random_author_amount_of_comments_iama_randomized =  
author\_information\_random\_sampleset\_randomized['amount_of_comments_iama']
```

Definition at line 185 of file a__everything_Big_CSV_analyzer.py.

```
a__everything_Big_CSV_analyzer.random_author_author_birth_date =  
author\_information\_random['author_birth_date']
```

Definition at line 168 of file a__everything_Big_CSV_analyzer.py.

```
a__everything_Big_CSV_analyzer.random_author_author_birth_date_randomized =  
author\_information\_random\_sampleset\_randomized['author_birth_date']
```

Definition at line 186 of file a__everything_Big_CSV_analyzer.py.

```
a__everything_Big_CSV_analyzer.random_author_author_comment_karma_amount =  
author\_information\_random['author_comment_karma_amount']
```

Definition at line 169 of file a__everything_Big_CSV_analyzer.py.

a__everything_Big_CSV_analyzer.random_author_author_comment_karma_amount_randomized = [author_information_random_sampleset_randomized](#)['author_comment_karma_amount']

Definition at line 187 of file a__everything_Big_CSV_analyzer.py.

a__everything_Big_CSV_analyzer.random_author_author_link_karma_amount = [author_information_random](#)['author_link_karma_amount']

Definition at line 170 of file a__everything_Big_CSV_analyzer.py.

a__everything_Big_CSV_analyzer.random_author_author_link_karma_amount_randomized = [author_information_random_sampleset_randomized](#)['author_link_karma_amount']

Definition at line 188 of file a__everything_Big_CSV_analyzer.py.

a__everything_Big_CSV_analyzer.random_author_author_name = [author_information_random](#)['author_name']

Definition at line 171 of file a__everything_Big_CSV_analyzer.py.

a__everything_Big_CSV_analyzer.random_author_author_name_randomized = [author_information_random_sampleset_randomized](#)['author_name']

Definition at line 189 of file a__everything_Big_CSV_analyzer.py.

a__everything_Big_CSV_analyzer.random_author_comment_creation_every_x_sec = [author_information_random](#)['comment_creation_every_x_sec']

Definition at line 172 of file a__everything_Big_CSV_analyzer.py.

a__everything_Big_CSV_analyzer.random_author_comment_creation_every_x_sec_randomized = [author_information_random_sampleset_randomized](#)['comment_creation_every_x_sec']

Definition at line 190 of file a__everything_Big_CSV_analyzer.py.

a__everything_Big_CSV_analyzer.random_author_thread_creation_every_x_sec = [author_information_random](#)['thread_creation_every_x_sec']

Definition at line 173 of file a__everything_Big_CSV_analyzer.py.

a__everything_Big_CSV_analyzer.random_author_thread_creation_every_x_sec_randomized = [author_information_random_sampleset_randomized](#)['thread_creation_every_x_sec']

Definition at line 191 of file a__everything_Big_CSV_analyzer.py.

a__everything_Big_CSV_analyzer.random_author_time_acc_birth_first_iama_thread = [author_information_random](#)['time_acc_birth_first_iama_thread']

Definition at line 174 of file a__everything_Big_CSV_analyzer.py.

a__everything_Big_CSV_analyzer.random_author_time_acc_birth_first_iama_thread_randomized = [author_information_random_sampleset_randomized](#)['time_acc_birth_first_iama_thread']

Definition at line 192 of file a__everything_Big_CSV_analyzer.py.

**a__everything_Big_CSV_analyzer.random_author_time_diff_acc_creation_n_first_comment = **

Definition at line 175 of file a__everything_Big_CSV_analyzer.py.

**a__everything_Big_CSV_analyzer.random_author_time_diff_acc_creation_n_first_comment_randomized = **

Definition at line 193 of file a__everything_Big_CSV_analyzer.py.

a__everything_Big_CSV_analyzer.random_author_time_diff_acc_creation_n_first_thread

```
Initial value: 1 = author_information_random[
2             'time_diff_acc_creation_n_first_thread']
```

Definition at line 177 of file a__everything_Big_CSV_analyzer.py.

a__everything_Big_CSV_analyzer.random_author_time_diff_acc_creation_n_first_thread_randomized

```
Initial value: 1 = author_information_random_sampleset_randomized[
2             'time_diff_acc_creation_n_first_thread']
```

Definition at line 195 of file a__everything_Big_CSV_analyzer.py.

a__everything_Big_CSV_analyzer.thread_amount_of_commentators_tier_1

```
Initial value: 1 = thread_information[
2             'Thread amount of commentators tier 1']
```

Definition at line 286 of file a__everything_Big_CSV_analyzer.py.

a__everything_Big_CSV_analyzer.thread_amount_of_commentators_tier_x

```
Initial value: 1 = thread_information[
2             'Thread amount of commentators tier x']
```

Definition at line 288 of file a__everything_Big_CSV_analyzer.py.

a__everything_Big_CSV_analyzer.thread_amount_of_commentators_total

```
Initial value: 1 = thread_information[
2             'Thread amount of commentators total']
```

Definition at line 284 of file a__everything_Big_CSV_analyzer.py.

a__everything_Big_CSV_analyzer.thread_amount_of_questioners_tier_1

```
Initial value: 1 = thread_information[
2             'Thread amount of questioners tier 1']
```

Definition at line 279 of file a__everything_Big_CSV_analyzer.py.

a__everything_Big_CSV_analyzer.thread_amount_of_questioners_tier_x

```
Initial value: 1 = thread_information[
2      'Thread amount of questioners tier x']
```

Definition at line 281 of file a__everything_Big_CSV_analyzer.py.

a__everything_Big_CSV_analyzer.thread_amount_of_questioners_total

```
Initial value: 1 = thread_information[
2      'Thread amount of questioners total']
```

Definition at line 277 of file a__everything_Big_CSV_analyzer.py.

a__everything_Big_CSV_analyzer.thread_author = [thread_information](#)['Thread author']

Definition at line 205 of file a__everything_Big_CSV_analyzer.py.

a__everything_Big_CSV_analyzer.thread_average_comment_vote_score_tier_1

```
Initial value: 1 = thread_information[
2      'Thread average comment vote score tier 1']
```

Definition at line 213 of file a__everything_Big_CSV_analyzer.py.

a__everything_Big_CSV_analyzer.thread_average_comment_vote_score_tier_x

```
Initial value: 1 = thread_information[
2      'Thread average comment vote score tier x']
```

Definition at line 215 of file a__everything_Big_CSV_analyzer.py.

a__everything_Big_CSV_analyzer.thread_average_comment_vote_score_total

```
Initial value: 1 = thread_information[
2      'Thread average comment vote score total']
```

Definition at line 210 of file a__everything_Big_CSV_analyzer.py.

a__everything_Big_CSV_analyzer.thread_average_question_vote_score_tier_1

```
Initial value: 1 = thread_information[
2      'Thread average question vote score tier 1']
```

Definition at line 220 of file a__everything_Big_CSV_analyzer.py.

a__everything_Big_CSV_analyzer.thread_average_question_vote_score_tier_x

```
Initial value: 1 = thread_information[
2      'Thread average question vote score tier x']
```

Definition at line 222 of file a__everything_Big_CSV_analyzer.py.

a__everything_Big_CSV_analyzer.thread_average_question_vote_score_total

```
Initial value: 1 = thread_information[
2      'Thread average question vote score total']
```

Definition at line 218 of file a__everything_Big_CSV_analyzer.py.

a__everything_Big_CSV_analyzer.thread_average_reaction_time_between_comments_tier_1

```
Initial value: 1 = thread_information[
2      'Thread average reaction time between comments tier 1']
```

Definition at line 251 of file a__everything_Big_CSV_analyzer.py.

a__everything_Big_CSV_analyzer.thread_average_reaction_time_between_comments_tier_x

```
Initial value: 1 = thread_information[
2      'Thread average reaction time between comments tier x']
```

Definition at line 253 of file a__everything_Big_CSV_analyzer.py.

a__everything_Big_CSV_analyzer.thread_average_reaction_time_between_comments_total

```
Initial value: 1 = thread_information[
2 'Thread average reaction time between comments total']
```

Definition at line 249 of file a__everything_Big_CSV_analyzer.py.

a__everything_Big_CSV_analyzer.thread_average_reaction_time_between_questions_tier_1

```
Initial value: 1 = thread_information[
2 'Thread average reaction time between questions tier 1']
```

Definition at line 258 of file a__everything_Big_CSV_analyzer.py.

a__everything_Big_CSV_analyzer.thread_average_reaction_time_between_questions_tier_x

```
Initial value: 1 = thread_information[
2 'Thread average reaction time between questions tier x']
```

Definition at line 260 of file a__everything_Big_CSV_analyzer.py.

a__everything_Big_CSV_analyzer.thread_average_reaction_time_between_questions_total

```
Initial value: 1 = thread_information[
2 'Thread average reaction time between questions total']
```

Definition at line 256 of file a__everything_Big_CSV_analyzer.py.

a__everything_Big_CSV_analyzer.thread_average_response_to_comment_time_iama_host_tier_1

```
Initial value: 1 = thread_information[
2 'Thread average response to comment time iama host tier 1']
```

Definition at line 265 of file a__everything_Big_CSV_analyzer.py.

a__everything_Big_CSV_analyzer.thread_average_response_to_comment_time_iama_host_tier_x

```
Initial value: 1 = thread_information[
2 'Thread average response to comment time iama host tier x']
```

Definition at line 267 of file a__everything_Big_CSV_analyzer.py.

a__everything_Big_CSV_analyzer.thread_average_response_to_comment_time_iama_host_total

```
Initial value: 1 = thread_information[
2 'Thread average response to comment time iama host total']
```

Definition at line 263 of file a__everything_Big_CSV_analyzer.py.

a__everything_Big_CSV_analyzer.thread_average_response_to_question_time_iama_host_tier_1

```
Initial value: 1 = thread_information[
2 'Thread average response to question time iama host tier 1']
```

Definition at line 272 of file a__everything_Big_CSV_analyzer.py.

a__everything_Big_CSV_analyzer.thread_average_response_to_question_time_iama_host_tier_x

```
Initial value: 1 = thread_information[
2 'Thread average response to question time iama host tier x']
```

Definition at line 274 of file a__everything_Big_CSV_analyzer.py.

a__everything_Big_CSV_analyzer.thread_average_response_to_question_time_iama_host_total

```
Initial value: 1 = thread_information[
2 'Thread average response to question time iama host total']
```

Definition at line 270 of file a__everything_Big_CSV_analyzer.py.

a__everything_Big_CSV_analyzer.thread_creation_time_stamp = [thread_information](#)['Thread creation time stamp']

Definition at line 208 of file a__everything_Big_CSV_analyzer.py.

a__everything_Big_CSV_analyzer.thread_downs = [thread_information](#)['Thread downs']

Definition at line 207 of file a__everything_Big_CSV_analyzer.py.

a__everything_Big_CSV_analyzer.thread_id = [thread_information](#)['Thread id']

Definition at line 204 of file a__everything_Big_CSV_analyzer.py.

a__everything_Big_CSV_analyzer.thread_information

```
Initial value: 1 = pandas.read_csv(  
2     'all.csv',  
3     sep=',',  
4     na_values="None")
```

Definition at line 104 of file a__everything_Big_CSV_analyzer.py.

a__everything_Big_CSV_analyzer.thread_life_span_until_last_comment

```
Initial value: 1 = thread_information[  
2     'Thread life span until last comment']
```

Definition at line 292 of file a__everything_Big_CSV_analyzer.py.

a__everything_Big_CSV_analyzer.thread_life_span_until_last_question

```
Initial value: 1 = thread_information[  
2     'Thread life span until last question']
```

Definition at line 294 of file a__everything_Big_CSV_analyzer.py.

a__everything_Big_CSV_analyzer.thread_num_comments_answered_by_iam_a_host_tier_1

```
Initial value: 1 = thread_information[  
2     'Thread num comments answered by iama host tier 1']
```

Definition at line 244 of file a__everything_Big_CSV_analyzer.py.

a__everything_Big_CSV_analyzer.thread_num_comments_answered_by_iam_a_host_tier_x

```
Initial value: 1 = thread_information[  
2     'Thread num comments answered by iama host tier x']
```

Definition at line 246 of file a__everything_Big_CSV_analyzer.py.

a__everything_Big_CSV_analyzer.thread_num_comments_answered_by_iam_a_host_total

```
Initial value: 1 = thread_information[  
2     'Thread num comments answered by iama host total']
```

Definition at line 242 of file a__everything_Big_CSV_analyzer.py.

a__everything_Big_CSV_analyzer.thread_num_comments_tier_1 = [thread_information](#)['Thread num comments tier 1']

Definition at line 228 of file a__everything_Big_CSV_analyzer.py.

a__everything_Big_CSV_analyzer.thread_num_comments_tier_x = [thread_information](#)['Thread num comments tier x']

Definition at line 229 of file a__everything_Big_CSV_analyzer.py.

a__everything_Big_CSV_analyzer.thread_num_comments_total = [thread_information](#)['Thread num comments total']

Definition at line 227 of file a__everything_Big_CSV_analyzer.py.

a__everything_Big_CSV_analyzer.thread_num_comments_total_skewed

```
Initial value: 1 = thread_information[
                2 'Thread num comments total skewed']
```

Definition at line 225 of file a__everything_Big_CSV_analyzer.py.

a__everything_Big_CSV_analyzer.thread_num_questions_answered_by_iam_a_host_tier_1

```
Initial value: 1 = thread_information[
                2 'Thread num questions answered by iama host tier 1']
```

Definition at line 237 of file a__everything_Big_CSV_analyzer.py.

a__everything_Big_CSV_analyzer.thread_num_questions_answered_by_iam_a_host_tier_x

```
Initial value: 1 = thread_information[
                2 'Thread num questions answered by iama host tier x']
```

Definition at line 239 of file a__everything_Big_CSV_analyzer.py.

a__everything_Big_CSV_analyzer.thread_num_questions_answered_by_iam_a_host_total

```
Initial value: 1 = thread_information[
                2 'Thread num questions answered by iama host total']
```

Definition at line 235 of file a__everything_Big_CSV_analyzer.py.

a__everything_Big_CSV_analyzer.thread_num_questions_tier_1 = [thread_information](#)['Thread num questions tier 1']

Definition at line 232 of file a__everything_Big_CSV_analyzer.py.

a__everything_Big_CSV_analyzer.thread_num_questions_tier_x = [thread_information](#)['Thread num questions tier x']

Definition at line 233 of file a__everything_Big_CSV_analyzer.py.

a__everything_Big_CSV_analyzer.thread_num_questions_total = [thread_information](#)['Thread num questions total']

Definition at line 231 of file a__everything_Big_CSV_analyzer.py.

a__everything_Big_CSV_analyzer.thread_ups = [thread_information](#)['Thread ups']

Definition at line 206 of file a__everything_Big_CSV_analyzer.py.

a__everything_Big_CSV_analyzer.thread_year = [thread_information](#)['Year']

Definition at line 203 of file a__everything_Big_CSV_analyzer.py.

a_author_Information Namespace Reference

Functions

- def [check_script_arguments](#) ()
- def [initialize_mongo_db_parameters](#) ()
- def [write_csv_data](#) ()

Variables

- [mongo_db_client_instance](#) = None
- [mongo_db_author_instance](#) = None
- [mongo_db_author_collection](#) = None
- int [mongo_db_author_collection_original](#) = 0
- string [argument_db_to_choose](#) = ""

Function Documentation

def a_author_Information.check_script_arguments ()

```
Checks if enough and correct arguments have been given to run this script adequate

1. It checks in the first instance if enough arguments have been given
2. Then necessary variables will be filled with appropriate values

Args:
-
Returns:
-
```

Definition at line 14 of file a_author_Information.py.

def a_author_Information.initialize_mongo_db_parameters ()

```
Instantiates all necessary variables for the correct usage of the mongoDB client

Args:
-
Returns:
-
```

Definition at line 48 of file a_author_Information.py.

def a_author_Information.write_csv_data ()

```
Gets all information from every collection within 'iAMA Reddit Authors*' database and writes it
into a csv file

Args:
-
Returns:
-
```

Definition at line 76 of file a_author_Information.py.

Variable Documentation

string a_author_Information.argument_db_to_choose = ""

Definition at line 177 of file a_author_Information.py.

a_author_Information.mongo_db_author_collection = None

Definition at line 169 of file a_author_Information.py.

int a_author_Information.mongo_db_author_collection_original = 0

Definition at line 173 of file a_author_Information.py.

a_author_Information.mongo_db_author_instance = None

Definition at line 166 of file a_author_Information.py.

a_author_Information.mongo_db_client_instance = None

Definition at line 163 of file a_author_Information.py.

a_iAMA_Commenttime_arr Namespace Reference

Functions

- def [check_script_arguments](#) ()
- def [initialize_mongo_db_parameters](#) (actually_processed_year)
- def [start_data_generation_for_analysis](#) ()
- def [prepare_data_for_graph](#) ()
- def [add_thread_list_to_global_list](#) (list_to_append)
- def [generate_data_to_be_analyzed](#) ()
- def [calculate_ar_mean_answer_time_for_questions](#) (id_of_thread, author_of_thread)
- def [check_if_comment_is_a_question](#) (given_string)
- def [check_if_comment_is_on_tier_1](#) (comment_parent_id)
- def [check_if_comment_is_not_from_thread_author](#) (author_of_thread, comment_author)
- def [check_if_comment_is_answer_from_thread_author](#) (author_of_thread, comment_actual_id, comments_cursor)
- def [calculate_time_difference](#) (comment_time_stamp, answer_time_stamp_iama_host)
- def [write_csv_data](#) (list_with_information)
- def [plot_generated_data](#) ()

Variables

- int [argument_year_beginning](#) = 0
- int [year_actually_in_progress](#) = 0
- int [argument_year_ending](#) = 0
- string [argument_tier_in_scope](#) = ""
- string [argument_plot_time_unit](#) = ""
- [mongo_DB_Client_Instance](#) = None
- [mongo_DB_Threads_Instance](#) = None
- [mongo_DB_Thread_Collection](#) = None
- [mongo_DB_Comments_Instance](#) = None
- list [list_To_Be_Plotted](#) = []
- list [global_thread_list](#) = []
- list [data_to_give_plotly](#) = []

Function Documentation

def a_iAMA_Commenttime_arr.add_thread_list_to_global_list (*list_to_append*)

```
Adds all elements of for the current year into a global list. This global list will be written into
a csv file
later on

1. This method simply checks whether both strings match each other or not.
   I have built this extra method to have a better overview in the main code..

Args:
    list_to_append (list) : The list which will be iterated over and which elements will be added
to the global list
Returns:
    -
```

Definition at line 248 of file a_iAMA_Commenttime_arr.py.


```
def a_iAMA_Commenttime_arr.calculate_ar_mean_answer_time_for_questions ( id_of_thread,
author_of_thread)
```

```
Calculates the arithmetic mean of the answer time by the iama host in minutes

In dependence of the given tier argument (second argument) the processing of tiers will be filtered

Args:
    id_of_thread (str): The id of the thread which is actually processed. (Necessary for checking
    if a question
        lies on tier 1 or any other tier)
    author_of_thread (str): The name of the thread author. (Necessary for checking if a given answer
    is from the
        iama host or not)
Returns:
    Whenever there was a minimum of 1 question asked and 1 answer from the iama host:
        amount of answer times (int) : The amount of the arithmetic mean time of
    Whenever there no questions have been asked for that thread / or no answers were given /
    or all values in the database were null:
        None: Returns an empty object of the type None
```

Definition at line 315 of file a_iAMA_Commenttime_arr.py.

```
def a_iAMA_Commenttime_arr.calculate_time_difference ( comment_time_stamp,
answer_time_stamp_iama_host)
```

```
Calculates the time difference in seconds between the a comment and its answer from the iama host

1. The time stamps will be converted from epoch
    into float and afterwards into str again
    (necessary for correct subtraction)
2. Then the time stamps will be subtracted from each other
3. The containing time difference will be converted into seconds (int)

Args:
    comment_time_stamp (str): The time stamp of the comment
    answer_time_stamp_iama_host (str): The time stamp of the iAMA hosts answer
Returns:
    time difference in seconds (int) : The time difference of the comment and its answer by the
    iAMA host in seconds
```

Definition at line 593 of file a_iAMA_Commenttime_arr.py.

```
def a_iAMA_Commenttime_arr.check_if_comment_is_a_question ( given_string)
```

```
Simply checks whether a given string is a question or not

1. This method simply checks whether a question mark exists within that string or not..
    This is just that simple because messing around with natural processing kits to determine the
    semantic sense
        would blow up my bachelor work...

Args:
    given_string (int) : The string which will be checked for a question mark
Returns:
    True (bool): Whenever the given string is a question
    False (bool): Whenever the given string is not a question
```

Definition at line 491 of file a_iAMA_Commenttime_arr.py.

```
def a_iAMA_Commenttime_arr.check_if_comment_is_answer_from_thread_author (  
author_of_thread, comment_actual_id, comments_cursor)
```

```
Checks whether both strings are equal or not

1. A dictionary containing flags whether that a question is answered by the host with the appropriate
timestamp will
    be created in the beginning.
2. Then the method iterates over every comment within that thread
    1.1. Whenever an answer is from the iAMA hosts and the processed comments 'parent_id' matches
the iAMA hosts
        comments (answers) id, the returned dict will contain appropriate values and will be
returned
    1.2. If this is not the case, it will be returned in its default condition

Args:
author of thread (str) : The name of the thread author (iAMA-Host)
comment actual id (str) : The id of the actually processed comment
comments cursor (list) : The cursor which shows to the amount of comments which can be iterated

Returns:
True (bool): Whenever the strings do not match
False (bool): Whenever the strings do match
answered that given question)
```

Definition at line 548 of file a_iAMA_Commenttime_arr.py.

```
def a_iAMA_Commenttime_arr.check_if_comment_is_not_from_thread_author (  
author_of_thread, comment_author)
```

```
Checks whether both strings are equal or not

1. This method simply checks whether both strings match each other or not.
    I have built this extra method to have a better overview in the main code..

Args:
author_of_thread (str) : The name of the thread author (iAMA-Host)
comment_author (str) : The name of the comments author

Returns:
True (bool): Whenever the strings do not match
False (bool): Whenever the strings do match
answered that given question)
```

Definition at line 528 of file a_iAMA_Commenttime_arr.py.

```
def a_iAMA_Commenttime_arr.check_if_comment_is_on_tier_1 ( comment_parent_id)
```

```
Checks whether a comment relies on the first tier or any other tier

Args:
comment parent id (str) : The name id of the comments parent

Returns:
True (bool): Whenever the comment lies on tier 1
False (bool): Whenever the comment lies on any other tier
```

Definition at line 512 of file a_iAMA_Commenttime_arr.py.

```
def a_iAMA_Commenttime_arr.check_script_arguments ()
```

```
Checks if enough and correct arguments have been given to run this script adequate
```

```
1. It checks in the first instance if enough arguments have been given
2. Then necessary variables will be filled with appropriate values
```

```
Args:
-
Returns:
-
```

Definition at line 20 of file a_iAMA_Commenttime_arr.py.

def a_iAMA_Commenttime_arr.generate_data_to_be_analyzed ()

```
Generates the data which will be analyzed
```

```
1. This method iterates over every thread
  1.1. It filters if that iterated thread is an iAMA-request or not
      1.1.1. If yes: this thread gets skipped and the next one will be processed
      1.1.2. If no: this thread will be processed
2. If the thread gets processed it will receive the arithmetic mean of answer time
3. This value will be added to a global list and will be plotted later on
```

```
Args:
-
Returns:
-
```

Definition at line 267 of file a_iAMA_Commenttime_arr.py.

def a_iAMA_Commenttime_arr.initialize_mongo_db_parameters (*actually_processed_year*)

```
Instantiates all necessary variables for the correct usage of the mongoDB-Client
```

```
Args:
  actually_processed_year (int) : The year with which parameters the database should be accessed
Returns:
-
```

Definition at line 47 of file a_iAMA_Commenttime_arr.py.

def a_iAMA_Commenttime_arr.plot_generated_data ()

```
Plots the data which is to be generated
```

```
1. This method plots the data which has been calculated before by using Pltoly-Framework within
a self written class
```

```
Args:
-
Returns:
-
```

Definition at line 691 of file a_iAMA_Commenttime_arr.py.

def a_iAMA_Commenttime_arr.prepare_data_for_graph ()

```
Sorts and prepares data for graph plotting
```

```
Args:
-
```

```
Returns:  
-
```

Definition at line 147 of file a_iAMA_Commenttime_arr.py.

def a_iAMA_Commenttime_arr.start_data_generation_for_analysis ()

```
Starts the data processing by swichting through the years  
  
1. Triggers the data generation process and moves forward within the years  
    1.1. By moving through the years a csv file will be created for every year  
    1.2. Additionally an interactive chart will be plotted  
  
Args:  
-  
Returns:  
-
```

Definition at line 67 of file a_iAMA_Commenttime_arr.py.

def a_iAMA_Commenttime_arr.write_csv_data (*list_with_information*)

```
Creates a csv file containing all necessary information about the average comment time of the iama  
host  
  
Args:  
    list with information (list) : Contains various information about thread and comment time  
Returns:  
-
```

Definition at line 632 of file a_iAMA_Commenttime_arr.py.

Variable Documentation

string a_iAMA_Commenttime_arr.argument_plot_time_unit = ""

Definition at line 720 of file a_iAMA_Commenttime_arr.py.

string a_iAMA_Commenttime_arr.argument_tier_in_scope = ""

Definition at line 717 of file a_iAMA_Commenttime_arr.py.

int a_iAMA_Commenttime_arr.argument_year_beginning = 0

Definition at line 708 of file a_iAMA_Commenttime_arr.py.

int a_iAMA_Commenttime_arr.argument_year_ending = 0

Definition at line 714 of file a_iAMA_Commenttime_arr.py.

list a_iAMA_Commenttime_arr.data_to_give_plotly = []

Definition at line 752 of file a_iAMA_Commenttime_arr.py.

list a_iAMA_Commenttime_arr.global_thread_list = []

Definition at line 738 of file a_iAMA_Commenttime_arr.py.

list a_iAMA_Commenttime_arr.list_To_Be_Plotted = []

Definition at line 735 of file a_iAMA_Commenttime_arr.py.

a_iAMA_Commenttime_arr.mongo_DB_Client_Instance = None

Definition at line 723 of file a_iAMA_Commenttime_arr.py.

a_iAMA_Commenttime_arr.mongo_DB_Comments_Instance = None

Definition at line 732 of file a_iAMA_Commenttime_arr.py.

a_iAMA_Commenttime_arr.mongo_DB_Thread_Collection = None

Definition at line 729 of file a_iAMA_Commenttime_arr.py.

a_iAMA_Commenttime_arr.mongo_DB_Threads_Instance = None

Definition at line 726 of file a_iAMA_Commenttime_arr.py.

int a_iAMA_Commenttime_arr.year_actually_in_progress = 0

Definition at line 711 of file a_iAMA_Commenttime_arr.py.

a_iAMA_Commenttime_med Namespace Reference

Functions

- def [check_script_arguments](#) ()
- def [initialize_mongo_db_parameters](#) (actually_processed_year)
- def [start_data_generation_for_analysis](#) ()
- def [prepare_data_for_graph](#) ()
- def [add_thread_list_to_global_list](#) (list_to_append)
- def [generate_data_to_be_analyzed](#) ()
- def [calculate_ar_mean_answer_time_for_questions](#) (id_of_thread, author_of_thread)
- def [check_if_comment_is_a_question](#) (given_string)
- def [check_if_comment_is_on_tier_1](#) (comment_parent_id)
- def [check_if_comment_is_not_from_thread_author](#) (author_of_thread, comment_author)
- def [check_if_comment_is_answer_from_thread_author](#) (author_of_thread, comment_actual_id, comments_cursor)
- def [calculate_time_difference](#) (comment_time_stamp, answer_time_stamp_iama_host)
- def [write_csv_data](#) (list_with_information)
- def [plot_generated_data](#) ()

Variables

- int [argument_year_beginning](#) = 0
- int [year_actually_in_progress](#) = 0
- int [argument_year_ending](#) = 0
- string [argument_tier_in_scope](#) = ""
- string [argument_plot_time_unit](#) = ""
- [mongo_DB_Client_Instance](#) = None
- [mongo_DB_Threads_Instance](#) = None
- [mongo_DB_Thread_Collection](#) = None
- [mongo_DB_Comments_Instance](#) = None
- list [list_To_Be_Plotted](#) = []
- list [global_thread_list](#) = []
- list [data_to_give_plotly](#) = []

Function Documentation

def a_iAMA_Commenttime_med.add_thread_list_to_global_list (*list_to_append*)

```
Adds all elements of for the current year into a global list. This global list will be written into
a csv file
later on

1. This method simply checks whether both strings match each other or not.
   I have built this extra method to have a better overview in the main code..

Args:
    list_to_append (list) : The list which will be iterated over and which elements will be added
to the global list
Returns:
    -
```

Definition at line 248 of file a_iAMA_Commenttime_med.py.

```
def a_iAMA_Commenttime_med.calculate_ar_mean_answer_time_for_questions ( id_of_thread,
author_of_thread)
```

```
Calculates the arithmetic mean of the answer time by the iama host in minutes

In dependence of the given tier argument (second argument) the processing of tiers will be filtered

Args:
    id_of_thread (str): The id of the thread which is actually processed. (Necessary for checking
if a question
        lies on tier 1 or any other tier)
    author_of_thread (str): The name of the thread author. (Necessary for checking if a given answer
is from the
        iama host or not)
Returns:
    Whenever there was a minimum of 1 question asked and 1 answer from the iama host:
        amount of answer times (int) : The amount of the arithmetic mean time of
    Whenever there no questions have been asked for that thread / or no answers were given /
    or all values in the database were null:
        None: Returns an empty object of the type None
```

Definition at line 315 of file a_iAMA_Commenttime_med.py.

```
def a_iAMA_Commenttime_med.calculate_time_difference ( comment_time_stamp,
answer_time_stamp_iama_host)
```

```
Calculates the time difference in seconds between the a comment and its answer from the iama host

1. The time stamps will be converted from epoch
    into float and afterwards into str again
    (necessary for correct subtraction)
2. Then the time stamps will be subtracted from each other
3. The containing time difference will be converted into seconds (int)

Args:
    comment_time_stamp (str): The time stamp of the comment
    answer_time_stamp_iama_host (str): The time stamp of the iAMA hosts answer
Returns:
    time difference in seconds (int) : The time difference of the comment and its answer by the
iAMA host in seconds
```

Definition at line 593 of file a_iAMA_Commenttime_med.py.

```
def a_iAMA_Commenttime_med.check_if_comment_is_a_question ( given_string)
```

```
Simply checks whether a given string is a question or not

1. This method simply checks whether a question mark exists within that string or not..
    This is just that simple because messing around with natural processing kits to determine the
semantic sense
    would blow up my bachelor work...

Args:
    given_string (int) : The string which will be checked for a question mark
Returns:
    True (bool): Whenever the given string is a question
    False (bool): Whenever the given string is not a question
```

Definition at line 491 of file a_iAMA_Commenttime_med.py.

```
def a_iAMA_Commenttime_med.check_if_comment_is_answer_from_thread_author (  
author_of_thread, comment_actual_id, comments_cursor)
```

```
Checks whether both strings are equal or not

1. A dictionary containing flags whether that a question is answered by the host with the appropriate
timestamp will
    be created in the beginning.
2. Then the method iterates over every comment within that thread
    1.1. Whenever an answer is from the iAMA hosts and the processed comments 'parent_id' matches
the iAMA hosts
        comments (answers) id, the returned dict will contain appropriate values and will be
returned
    1.2. If this is not the case, it will be returned in its default condition

Args:
author of thread (str) : The name of the thread author (iAMA-Host)
comment actual id (str) : The id of the actually processed comment
comments cursor (list) : The cursor which shows to the amount of comments which can be iterated

Returns:
True (bool): Whenever the strings do not match
False (bool): Whenever the strings do match
answered that given question)
```

Definition at line 548 of file a_iAMA_Commenttime_med.py.

```
def a_iAMA_Commenttime_med.check_if_comment_is_not_from_thread_author (  
author_of_thread, comment_author)
```

```
Checks whether both strings are equal or not

1. This method simply checks whether both strings match each other or not.
    I have built this extra method to have a better overview in the main code..

Args:
author_of_thread (str) : The name of the thread author (iAMA-Host)
comment_author (str) : The name of the comments author

Returns:
True (bool): Whenever the strings do not match
False (bool): Whenever the strings do match
answered that given question)
```

Definition at line 528 of file a_iAMA_Commenttime_med.py.

```
def a_iAMA_Commenttime_med.check_if_comment_is_on_tier_1 ( comment_parent_id)
```

```
Checks whether a comment relies on the first tier or any other tier

Args:
comment parent id (str) : The name id of the comments parent

Returns:
True (bool): Whenever the comment lies on tier 1
False (bool): Whenever the comment lies on any other tier
```

Definition at line 512 of file a_iAMA_Commenttime_med.py.

```
def a_iAMA_Commenttime_med.check_script_arguments ()
```

```
Checks if enough and correct arguments have been given to run this script adequate
```



```
1. It checks in the first instance if enough arguments have been given
2. Then necessary variables will be filled with appropriate values
```

```
Args:
-
Returns:
-
```

Definition at line 20 of file a_iAMA_Commenttime_med.py.

def a_iAMA_Commenttime_med.generate_data_to_be_analyzed ()

```
Generates the data which will be analyzed
```

```
1. This method iterates over every thread
  1.1. It filters if that iterated thread is an iAMA-request or not
    1.1.1. If yes: this thread gets skipped and the next one will be processed
    1.1.2. If no: this thread will be processed
2. If the thread gets processed it will receive the arithmetic mean of answer time
3. This value will be added to a global list and will be plotted later on
```

```
Args:
-
Returns:
-
```

Definition at line 267 of file a_iAMA_Commenttime_med.py.

def a_iAMA_Commenttime_med.initialize_mongo_db_parameters (*actually_processed_year*)

```
Instantiates all necessary variables for the correct usage of the mongoDB-Client
```

```
Args:
  actually_processed_year (int) : The year with which parameters the database should be accessed
Returns:
-
```

Definition at line 47 of file a_iAMA_Commenttime_med.py.

def a_iAMA_Commenttime_med.plot_generated_data ()

```
Plots the data which is to be generated
```

```
1. This method plots the data which has been calculated before by using Pltoly-Framework within
a self written class
```

```
Args:
-
Returns:
-
```

Definition at line 691 of file a_iAMA_Commenttime_med.py.

def a_iAMA_Commenttime_med.prepare_data_for_graph ()

```
Sorts and prepares data for graph plotting
```

```
Args:
-
```

```
Returns:  
-
```

Definition at line 147 of file a_iAMA_Commenttime_med.py.

def a_iAMA_Commenttime_med.start_data_generation_for_analysis ()

```
Starts the data processing by swichting through the years  
  
1. Triggers the data generation process and moves forward within the years  
    1.1. By moving through the years a csv file will be created for every year  
    1.2. Additionally an interactive chart will be plotted  
  
Args:  
-  
Returns:  
-
```

Definition at line 67 of file a_iAMA_Commenttime_med.py.

def a_iAMA_Commenttime_med.write_csv_data (*list_with_information*)

```
Creates a csv file containing all necessary information about the average comment time of the iama  
host  
  
Args:  
    list with information (list) : Contains various information about thread and comment time  
Returns:  
-
```

Definition at line 632 of file a_iAMA_Commenttime_med.py.

Variable Documentation

string a_iAMA_Commenttime_med.argument_plot_time_unit = ""

Definition at line 720 of file a_iAMA_Commenttime_med.py.

string a_iAMA_Commenttime_med.argument_tier_in_scope = ""

Definition at line 717 of file a_iAMA_Commenttime_med.py.

int a_iAMA_Commenttime_med.argument_year_beginning = 0

Definition at line 708 of file a_iAMA_Commenttime_med.py.

int a_iAMA_Commenttime_med.argument_year_ending = 0

Definition at line 714 of file a_iAMA_Commenttime_med.py.

list a_iAMA_Commenttime_med.data_to_give_plotly = []

Definition at line 752 of file a_iAMA_Commenttime_med.py.

list a_iAMA_Commenttime_med.global_thread_list = []

Definition at line 738 of file a_iAMA_Commenttime_med.py.

list a_iAMA_Commenttime_med.list_To_Be_Plotted = []

Definition at line 735 of file a_iAMA_Commenttime_med.py.

a_iAMA_Commenttime_med.mongo_DB_Client_Instance = None

Definition at line 723 of file a_iAMA_Commenttime_med.py.

a_iAMA_Commenttime_med.mongo_DB_Comments_Instance = None

Definition at line 732 of file a_iAMA_Commenttime_med.py.

a_iAMA_Commenttime_med.mongo_DB_Thread_Collection = None

Definition at line 729 of file a_iAMA_Commenttime_med.py.

a_iAMA_Commenttime_med.mongo_DB_Threads_Instance = None

Definition at line 726 of file a_iAMA_Commenttime_med.py.

int a_iAMA_Commenttime_med.year_actually_in_progress = 0

Definition at line 711 of file a_iAMA_Commenttime_med.py.

a_question_Answered_Yes_No_Extrema Namespace Reference

Functions

- def [check_script_arguments](#) ()
- def [initialize_mongo_db_parameters](#) (actually_processed_year)
- def [start_data_generation_for_analysis](#) ()
- def [generate_data_now](#) ()
- def [process_answered_questions_within_thread](#) (id_of_thread, author_of_thread, thread_creation_date)
- def [check_if_comment_is_a_question](#) (given_string)
- def [check_if_comment_is_not_from_thread_author](#) (author_of_thread, comment_author)
- def [check_if_comment_has_been_answered_by_thread_author](#) (author_of_thread, comment_acutal_id, comments_cursor)
- def [calculate_time_difference](#) (comment_time_stamp, answer_time_stamp_iama_host)
- def [sort_questions](#) (list_which_is_to_be_sorted)
- def [create_question_list_containing_all_years](#) (list_with_comments_per_years)
- def [write_csv_and_count_unanswered](#) (list_with_comments)
- def [plot_generated_data](#) ()

Variables

- int [argument_year_beginning](#) = 0
- int [year_actually_in_progress](#) = 0
- int [argument_year_ending](#) = 0
- [argument_sorting](#) = bool
- int [argument_amount_of_top_quotes](#) = 0
- [mongo_DB_Client_Instance](#) = None
- [mongo_DB_Threads_Instance](#) = None
- [mongo_DB_Thread_Collection](#) = None
- [mongo_DB_Comments_Instance](#) = None
- list [question_information_list](#) = []
- list [data_to_give_plotly](#) = []

Function Documentation

def a_question_Answered_Yes_No_Extrema.calculate_time_difference (*comment_time_stamp*, *answer_time_stamp_iama_host*)

```
Calculates the time difference in seconds between the a comment and its answer from the iama host

1. The time stamps will be converted from epoch
   into float and afterwards into str again
   (necessary for correct subtraction)
2. Then the time stamps will be subtracted from each other
3. The containing time difference will be converted into seconds (int)

Args:
    comment_time_stamp (str): The time stamp of the comment
    answer_time_stamp_iama_host (str): The time stamp of the iAMA hosts answer
Returns:
    time difference in seconds (int) : The time difference of the comment and its answer by the
    iAMA host in seconds
```

Definition at line 425 of file a_question_Answered_Yes_No_Extrema.py.

```
def
a_question_Answered_Yes_No_Extrema.check_if_comment_has_been_answered_by_thread_auth
or ( author_of_thread, comment_acutal_id, comments_cursor)
```

```
Checks whether both strings are equal or not

1. A dictionary containing flags whether that a question is answered by the host with the appropriate
timestamp will
    be created in the beginning.
2. Then the method iterates over every comment within that thread
    1.1. Whenever an answer is from the iAMA hosts and the processed comments 'parent id' matches
the iAMA hosts
        comments (answers) id, the returned dict will contain appropriate values and will be
returned
    1.2. If this is not the case, it will be returned in its default condition

Args:
author_of_thread (str) : The name of the thread author (iAMA-Host)
comment_acutal_id (str) : The id of the actually processed comment
comments cursor (list) : The cursor which shows to the amount of comments which can be iterated
Returns:
True (bool): Whenever the strings do not match
False (bool): Whenever the strings do match
answered that given question)
```

Definition at line 381 of file a_question_Answered_Yes_No_Extrema.py.

```
def a_question_Answered_Yes_No_Extrema.check_if_comment_is_a_question ( given_string)
```

```
Simply checks whether a given string is a question or not

1. This method simply checks whether a question mark exists within that string or not..
    This is just that simple because messing around with natural processing kits to determine the
semantic sense
    would blow up my bachelor work...

Args:
given_string (str) : The string which will be checked for a question mark
Returns:
True (bool): Whenever the given string is a question
False (bool): Whenever the given string is not a question
```

Definition at line 340 of file a_question_Answered_Yes_No_Extrema.py.

```
def a_question_Answered_Yes_No_Extrema.check_if_comment_is_not_from_thread_author (
author_of_thread, comment_author)
```

```
Checks whether both strings are equal or not

1. This method simply checks whether both strings match each other or not.
    I have built this extra method to have a better overview in the main code..

Args:
author of thread (str) : The name of the thread author (iAMA-Host)
comment_author (str) : The name of the comments author
Returns:
True (bool): Whenever the strings do not match
False (bool): Whenever the strings do match
answered that given question)
```

Definition at line 361 of file a_question_Answered_Yes_No_Extrema.py.

def a_question_Answered_Yes_No_Extrema.check_script_arguments ()

```
Checks if enough and correct arguments have been given to run this script adequate

1. It checks in the first instance if enough arguments have been given
2. Then necessary variables will be filled with appropriate values

Args:
-
Returns:
-
```

Definition at line 22 of file a_question_Answered_Yes_No_Extrema.py.

def a_question_Answered_Yes_No_Extrema.create_question_list_containing_all_years (list_with_comments_per_years)

```
Creates a list, containing all questions from all years

Args:
    list_with_comments_per_years (list) : The list containing the current years questions
Returns:
-
```

Definition at line 494 of file a_question_Answered_Yes_No_Extrema.py.

def a_question_Answered_Yes_No_Extrema.generate_data_now ()

```
Generates the data which will be written into csv and plotted later on

1. This method iterates over every thread
    1.1. It filters if that iterated thread is an iAMA-request or not
        1.1.1. If yes: this thread gets skipped and the next one will be processed
        1.1.2. If no: this thread will be processed
2. If the thread gets processed it will receive an ordered dictionary containing information about
every question
    whether it has been answered or not
3. This ordered dictionary will be appended to a global list, which will be processed afterwards
for the generation
    of plots and csv files

Args:
-
Returns:
-
```

Definition at line 165 of file a_question_Answered_Yes_No_Extrema.py.

def a_question_Answered_Yes_No_Extrema.initialize_mongo_db_parameters (actually_processed_year)

```
Instantiates all necessary variables for the correct usage of the mongoDB client

Args:
    actually_processed_year (int) : The year with which parameters the database should be accessed
Returns:
-
```

Definition at line 59 of file a_question_Answered_Yes_No_Extrema.py.

def a_question_Answered_Yes_No_Extrema.plot_generated_data ()

```
Plots the data which is to be generated

1. This method plots the data which has been calculated before by using Pltoly-Framework within
a self written class

Args:
-
Returns:
-
```

Definition at line 583 of file a_question_Answered_Yes_No_Extrema.py.

def a_question_Answered_Yes_No_Extrema.process_answered_questions_within_thread (***id_of_thread, author_of_thread, thread_creation_date*)**

```
Checks whether an iterated question has been answered by the iama host or not

1. This method checks at first whether an iterated comment contains values (e.g. is not none)
  1.1. If not: That comment will be skipped / if no comment is remaining None will be returned
  1.2. If yes: That comment will be processed
2. Now it will be checked whether that iterated comment is a question or not
3. Afterwards it will be checked whether that comment is a comment from the iAMA Host or not
  3.1. If this is not the case the next comment will be processed
4. Whenever that processed comment is a question and not (!!) from the thread author:
  amount_of_tier_any_questions (int) will be increased by one
5. Now it will be checked whether that comment has a comment ( answer ) below it which is from the
iAMA-host
  5.1. If yes: amount of tier any questions answered (int) will be increased by one and the
dictionary, which
  is to be returned will be filled with values
  5.2. If no: the dictionary, which is to be returned will be filled with values

Args:
id_of_thread (str) : Contains the id of the thread which is to be iterated
author_of_thread (str) : Contains the name of the thread author
thread_creation_date (str): Contains the time
Returns:
amount_of_questions_not_answered (int) : The amount of questions which have not been answered
```

Definition at line 217 of file a_question_Answered_Yes_No_Extrema.py.

def a_question_Answered_Yes_No_Extrema.sort_questions (*list_which_is_to_be_sorted*)

```
Sorts a list of questions for a year, depending on the upvotes

1. This method prepares the data, in kind of sorting and counting amount of questions not being
answered
2. It also returns the number of unanswered questions, necessary for chart plotting

Args:
list_which_is_to_be_sorted (list) : The list you want to sort regarding the sorting arguments
give on execution
Returns:
questions_sorted (list) : The amount of questions, sorted on upvotes
```

Definition at line 462 of file a_question_Answered_Yes_No_Extrema.py.

def a_question_Answered_Yes_No_Extrema.start_data_generation_for_analysis ()

```

Starts the data processing by swichting through the years

1. Triggers the data generation process and moves forward within the years
  1.1. By moving through the years a csv file will be created for every year
  1.2. At the end a csv file will be generated containing all questions of all years, sorted
  1.3. Additionally an interactive chart will be plotted

Args:
-
Returns:
-

```

Definition at line 79 of file `a_question_Answered_Yes_No_Extrema.py`.

`def a_question_Answered_Yes_No_Extrema.write_csv_and_count_unanswered (list_with_comments)`

```

Creates a csv file containing all necessary information and calculates the amount of unanswered
questions

1. This method iterates over the top / worst X comments
  1.1. By iterating: all necessary information will be written into the csv file
  1.2. By iterating: the amount of unanswered questions will be counted
2. After iterating the amount of unanswered questions will be returned, which is necessary for graph
plotting

Args:
  list_with_comments (list): Contains all comments from the year
Returns:
  amount_of_questions_not_answered (int) : The amount of questions which have not been answered

```

Definition at line 516 of file `a_question_Answered_Yes_No_Extrema.py`.

Variable Documentation

`int a_question_Answered_Yes_No_Extrema.argument_amount_of_top_quotes = 0`

Definition at line 613 of file `a_question_Answered_Yes_No_Extrema.py`.

`a_question_Answered_Yes_No_Extrema.argument_sorting = bool`

Definition at line 610 of file `a_question_Answered_Yes_No_Extrema.py`.

`int a_question_Answered_Yes_No_Extrema.argument_year_beginning = 0`

Definition at line 600 of file `a_question_Answered_Yes_No_Extrema.py`.

`int a_question_Answered_Yes_No_Extrema.argument_year_ending = 0`

Definition at line 606 of file `a_question_Answered_Yes_No_Extrema.py`.

`list a_question_Answered_Yes_No_Extrema.data_to_give_plotly = []`

Definition at line 642 of file a_question_Answered_Yes_No_Extrema.py.

a_question_Answered_Yes_No_Extrema.mongo_DB_Client_Instance = None

Definition at line 617 of file a_question_Answered_Yes_No_Extrema.py.

a_question_Answered_Yes_No_Extrema.mongo_DB_Comments_Instance = None

Definition at line 626 of file a_question_Answered_Yes_No_Extrema.py.

a_question_Answered_Yes_No_Extrema.mongo_DB_Thread_Collection = None

Definition at line 623 of file a_question_Answered_Yes_No_Extrema.py.

a_question_Answered_Yes_No_Extrema.mongo_DB_Threads_Instance = None

Definition at line 620 of file a_question_Answered_Yes_No_Extrema.py.

list a_question_Answered_Yes_No_Extrema.question_information_list = []

Definition at line 630 of file a_question_Answered_Yes_No_Extrema.py.

int a_question_Answered_Yes_No_Extrema.year_actually_in_progress = 0

Definition at line 603 of file a_question_Answered_Yes_No_Extrema.py.

a_question_Answered_Yes_No_Tier_Percentage Namespace Reference

Functions

- def [check_script_arguments](#) ()
- def [initialize_mongo_db_parameters](#) (actually_processed_year)
- def [start_data_generation_for_analysis](#) ()
- def [generate_data_to_be_analyzed](#) ()
- def [question_answering_distribution_tier1_tierx_tierany](#) (id_of_thread, author_of_thread)
- def [check_if_comment_is_a_question](#) (given_string)
- def [check_if_comment_is_on_tier_1](#) (comment_parent_id)
- def [check_if_comment_is_not_from_thread_author](#) (author_of_thread, comment_author)
- def [check_if_comment_is_answer_from_thread_author](#) (author_of_thread, comment_actual_id, comments_cursor)
- def [write_csv](#) (list_with_information)
- def [add_local_list_to_global_list](#) (list_to_append)
- def [prepare_data_for_graph](#) ()
- def [plot_generated_data](#) ()

Variables

- int [argument_year_beginning](#) = 0
- int [year_actually_in_progress](#) = 0
- int [argument_year_ending](#) = 0
- string [argument_tier_in_scope](#) = ""
- [mongo_DB_Client_Instance](#) = None
- [mongo_DB_Threads_Instance](#) = None
- [mongo_DB_Thread_Collection](#) = None
- [mongo_DB_Comments_Instance](#) = None
- list [global_question_list](#) = []
- list [year_question_list](#) = []
- list [data_to_give_plotly](#) = []

Function Documentation

def a_question_Answered_Yes_No_Tier_Percentage.add_local_list_to_global_list (list_to_append)

```
Adds all elements of for the current year into a global list. This global list will be written into a csv file later on
```

```
1. This method simply checks whether both strings match each other or not.
   I have built this extra method to have a better overview in the main code..
```

```
Args:
```

```
list_to_append (list) : The list which will be iterated over and which elements will be added to the global list
```

```
Returns:
```

```
-
```

Definition at line 483 of file a_question_Answered_Yes_No_Tier_Percentage.py.

```
def a_question_Answered_Yes_No_Tier_Percentage.check_if_comment_is_a_question (  
given_string)
```

```
Simply checks whether a given string is a question or not

1. This method simply checks whether a question mark exists within that string or not..
   This is just that simple because messing around with natural processing kits to determine the
   semantic sense
       would blow up my bachelor work...

Args:
    given_string (str) : The string which will be checked for a question mark
Returns:
    True (bool): Whenever the given string is a question
    False (bool): Whenever the given string is not a question
```

Definition at line 326 of file a_question_Answered_Yes_No_Tier_Percentage.py.

```
def  
a_question_Answered_Yes_No_Tier_Percentage.check_if_comment_is_answer_from_thread_auth  
or ( author_of_thread, comment_actual_id, comments_cursor)
```

```
Iterates over every comment, while comparing the ids and the comments creation author

1. the method iterates over every comment within that thread
   1.1. Whenever an answer is from the iAMA hosts and the processed comments 'parent id' matches
   the iAMA hosts
       comments (answers) id, the returned dict will contain appropriate values and will be
   returned
   1.2. If this is not the case, it will be returned in its default condition

Args:
    author_of_thread (str) : The name of the thread author (iAMA-Host)
    comment_actual_id: (str) : The id of the actually processed comment
    comments_cursor (Cursor) : The cursor which shows to the amount of comments which can be iterated
Returns:
    True (bool): Whenever the strings do not match
    False (bool): Whenever the strings do match
```

Definition at line 386 of file a_question_Answered_Yes_No_Tier_Percentage.py.

```
def  
a_question_Answered_Yes_No_Tier_Percentage.check_if_comment_is_not_from_thread_author (  
author_of_thread, comment_author)
```

```
Checks whether both strings are equal or not

1. This method simply checks whether both strings match each other or not.
   I have built this extra method to have a better overview in the main code..

Args:
    author_of_thread (str) : The name of the thread author (iAMA-Host)
    comment_author (str) : The name of the comments author
Returns:
    True (bool): Whenever the strings do not match
    False (bool): Whenever the strings do match
    answered that given question)
```

Definition at line 365 of file a_question_Answered_Yes_No_Tier_Percentage.py.

def a_question_Answered_Yes_No_Tier_Percentage.check_if_comment_is_on_tier_1 (comment_parent_id)

```
Simply checks whether a given string is a question posted on tier 1 or not

1. This method simply checks whether a question has been posted on tier 1 by looking whether the
given
    string contains the substring "t3_" or not

Args:
    comment parent id (str): The string which will be checked for "t3 " appearance in it
Returns:
    -
```

Definition at line 347 of file a_question_Answered_Yes_No_Tier_Percentage.py.

def a_question_Answered_Yes_No_Tier_Percentage.check_script_arguments ()

```
Checks if enough and correct arguments have been given to run this script adequate

1. It checks in the first instance if enough arguments have been given
2. Then necessary variables will be filled with appropriate values

Args:
    -
Returns:
    -
```

Definition at line 20 of file a_question_Answered_Yes_No_Tier_Percentage.py.

def a_question_Answered_Yes_No_Tier_Percentage.generate_data_to_be_analyzed ()

```
Generates the data which will be analyzed

1. This method iterates over every thread
    1.1. It filters if that iterated thread is an iAMA-request or not
        1.1.1. If yes: this thread gets skipped and the next one will be processed
        1.1.2. If no: this thread will be processed
2. If the thread gets processed it will receive the distribution of questions on the tiers
3. This value will be added to a global list and will be plotted later on

Args:
    -
Returns:
    -
```

Definition at line 141 of file a_question_Answered_Yes_No_Tier_Percentage.py.

def a_question_Answered_Yes_No_Tier_Percentage.initialize_mongo_db_parameters (actually_processed_year)

```
Instantiates all necessary variables for the correct usage of the mongoDB-Client

Args:
    actually_processed_year (int) : The year with which parameters the database should be accessed
Returns:
    -
```

Definition at line 45 of file a_question_Answered_Yes_No_Tier_Percentage.py.

def a_question_Answered_Yes_No_Tier_Percentage.plot_generated_data ()

```
Plots the data which is to be generated

1. This method plots the data which has been calculated before by using Pltoly-Framework within
a self written class

Args:
-
Returns:
-
```

Definition at line 528 of file a_question_Answered_Yes_No_Tier_Percentage.py.

def a_question_Answered_Yes_No_Tier_Percentage.prepare_data_for_graph ()

```
Sorts and prepares data for graph plotting

Args:
-
Returns:
-
```

Definition at line 502 of file a_question_Answered_Yes_No_Tier_Percentage.py.

def a_question_Answered_Yes_No_Tier_Percentage.question_answering_distribution_tier1_tierx_tier any (id_of_thread, author_of_thread)

```
Generates the data which will be analyzed

1. It iterates over every comment and
    1.1. checks if the iterated comment is a question
    1.2. checks if the iterated comment has been posted on tier 1 level
    1.3. checks if that comment is from the iAMA-Host himself or not

2. Now the posted question will be added to a global list, which will be used for csv writing and
chart generation
    later on

Args:
    id_of_thread (str) : Contains the id of the processed thread
    author_of_thread (str) : Contains the iAMA-Hosts name
Returns:
-
```

Definition at line 184 of file a_question_Answered_Yes_No_Tier_Percentage.py.

def a_question_Answered_Yes_No_Tier_Percentage.start_data_generation_for_analysis ()

```
Starts the data processing by swichting through the years

1. Triggers the data generation process and moves forward within the years
    1.1. By moving through the years a csv file will be created for every year
    1.2. Additionally an interactive chart will be plotted

Args:
-
Returns:
-
```

Definition at line 65 of file a_question_Answered_Yes_No_Tier_Percentage.py.

def a_question_Answered_Yes_No_Tier_Percentage.write_csv (*list_with_information*)

```
Creates a csv file containing all necessary information about the distribution of questions on the tiers

This method iterates over the the given list, which contains every single questions of that year (or all years) and writes a csv file containing misc information about those questions.

Args:
    list_with_information (list) : Contains various information about thread and comment time
Returns:
    -
```

Definition at line 417 of file a_question_Answered_Yes_No_Tier_Percentage.py.

Variable Documentation

string a_question_Answered_Yes_No_Tier_Percentage.argument_tier_in_scope = ""

Definition at line 556 of file a_question_Answered_Yes_No_Tier_Percentage.py.

int a_question_Answered_Yes_No_Tier_Percentage.argument_year_beginning = 0

Definition at line 547 of file a_question_Answered_Yes_No_Tier_Percentage.py.

int a_question_Answered_Yes_No_Tier_Percentage.argument_year_ending = 0

Definition at line 553 of file a_question_Answered_Yes_No_Tier_Percentage.py.

list a_question_Answered_Yes_No_Tier_Percentage.data_to_give_plotly = []

Definition at line 587 of file a_question_Answered_Yes_No_Tier_Percentage.py.

list a_question_Answered_Yes_No_Tier_Percentage.global_question_list = []

Definition at line 572 of file a_question_Answered_Yes_No_Tier_Percentage.py.

a_question_Answered_Yes_No_Tier_Percentage.mongo_DB_Client_Instance = None

Definition at line 560 of file a_question_Answered_Yes_No_Tier_Percentage.py.

a_question_Answered_Yes_No_Tier_Percentage.mongo_DB_Comments_Instance = None

Definition at line 569 of file a_question_Answered_Yes_No_Tier_Percentage.py.

a_question_Answered_Yes_No_Tier_Percentage.mongo_DB_Thread_Collection = None

Definition at line 566 of file a_question_Answered_Yes_No_Tier_Percentage.py.

a_question_Answered_Yes_No_Tier_Percentage.mongo_DB_Threads_Instance = None

Definition at line 563 of file a_question_Answered_Yes_No_Tier_Percentage.py.

int a_question_Answered_Yes_No_Tier_Percentage.year_actually_in_progress = 0

Definition at line 550 of file a_question_Answered_Yes_No_Tier_Percentage.py.

list a_question_Answered_Yes_No_Tier_Percentage.year_question_list = []

Definition at line 575 of file a_question_Answered_Yes_No_Tier_Percentage.py.

a_question_Tier_Distribution Namespace Reference

Functions

- def [initialize_mongo_db_parameters](#) (actually_processed_year)
- def [check_script_arguments](#) ()
- def [start_data_generation_for_analysis](#) ()
- def [generate_data_to_be_analyzed](#) ()
- def [question_distribution_tier1_tierx](#) (id_of_thread, author_of_thread)
- def [check_if_comment_is_a_question](#) (given_string)
- def [check_if_comment_is_on_tier_1](#) (comment_parent_id)
- def [check_if_comment_is_not_from_thread_author](#) (author_of_thread, comment_author)
- def [add_actual_year_list_to_global_list](#) (list_to_append)
- def [write_csv](#) (list_with_information)
- def [prepare_data_for_graph](#) ()
- def [plot_generated_data](#) ()

Variables

- int [argument_year_beginning](#) = 0
- int [year_actually_in_progress](#) = 0
- int [argument_year_ending](#) = 0
- [mongo_DB_Client_Instance](#) = None
- [mongo_DB_Threads_Instance](#) = None
- [mongo_DB_Thread_Collection](#) = None
- [mongo_DB_Comments_Instance](#) = None
- list [current_year_question_list](#) = []
- list [global_year_question_list](#) = []
- list [data_to_give_plotly](#) = []

Function Documentation

def a_question_Tier_Distribution.add_actual_year_list_to_global_list (*list_to_append*)

```
Iterates over a given list with thread information and adds every single element to a global list
The global list will be printed to csv in the end
```

```
Args:
```

```
list to append (list) : List with thread information which will be appended to a global list
```

```
Returns:
```

```
-
```

Definition at line 329 of file a_question_Tier_Distribution.py.

def a_question_Tier_Distribution.check_if_comment_is_a_question (*given_string*)

```
Simply checks whether a given string is a question or not
```

```
1. This method simply checks whether a question mark exists within that string or not..
```

```
This is just that simple because messing around with natural processing kits to determine the
semantic sense
```

```
would blow up my bachelor work...
```

```
Args:
```



```
given_string (str) : The string which will be checked for a question mark
Returns:
    True (bool): Whenever the given string is a question
    False (bool): Whenever the given string is not a question
```

Definition at line 269 of file a_question_Tier_Distribution.py.

def a_question_Tier_Distribution.check_if_comment_is_not_from_thread_author (
author_of_thread, comment_author)

```
Checks whether both strings are equal or not

1. This method simply checks whether both strings match each other or not.
   I have built this extra method to have a better overview in the main code..

Args:
    author_of_thread (str) : The name of the thread author (iAMA-Host)
    comment_author (str) : The name of the comments author
Returns:
    True (bool): Whenever the strings do not match
    False (bool): Whenever the strings do match
                  answered that given question)
```

Definition at line 308 of file a_question_Tier_Distribution.py.

def a_question_Tier_Distribution.check_if_comment_is_on_tier_1 (comment_parent_id)

```
Simply checks whether a given string is a question posted on tier 1 or not

1. This method simply checks whether a question has been posted on tier 1 by looking whether the
   given
   string contains the substring "t3_" or not

Args:
    comment_parent_id (str): The string which will be checked for "t3_" appearance in it
Returns:
    -
```

Definition at line 290 of file a_question_Tier_Distribution.py.

def a_question_Tier_Distribution.check_script_arguments ()

```
Checks if enough and correct arguments have been given to run this script adequately

1. It checks in the first instance if enough arguments have been given
2. Then necessary variables will be filled with appropriate values

Args:
    -
Returns:
    -
```

Definition at line 40 of file a_question_Tier_Distribution.py.

def a_question_Tier_Distribution.generate_data_to_be_analyzed ()

```
Generates the data which will be analyzed

1. This method iterates over every thread
   1.1. It filters if that iterated thread is an iAMA-request or not
```

```

        1.1.1. If yes: this thread gets skipped and the next one will be processed
        1.1.2. If no: this thread will be processed
    2. If the thread gets processed it will receive the distribution of questions on the tiers
    3. This value will be added to a global list and will be plotted later on

Args:
    -
Returns:
    -

```

Definition at line 143 of file `a_question_Tier_Distribution.py`.

`def a_question_Tier_Distribution.initialize_mongo_db_parameters (actually_processed_year)`

```

Instantiates all necessary variables for the correct usage of the mongoDB-Client

Args:
    actually_processed_year (int) : The year with which parameters the database should be accessed
Returns:
    -

```

Definition at line 20 of file `a_question_Tier_Distribution.py`.

`def a_question_Tier_Distribution.plot_generated_data ()`

```

Plots the data which is to be generated

1. This method plots the data which has been calculated before by using Pltoly-Framework within
a self written class

Args:
    -
Returns:
    -

```

Definition at line 437 of file `a_question_Tier_Distribution.py`.

`def a_question_Tier_Distribution.prepare_data_for_graph ()`

```

Sorts and prepares data for graph plotting

Args:
    -
Returns:
    -

```

Definition at line 412 of file `a_question_Tier_Distribution.py`.

**`def a_question_Tier_Distribution.question_distribution_tier1_tierx (id_of_thread,
author_of_thread)`**

```

Generates the data which will be analyzed

1. It iterates over every comment and
    1.1. checks if the iterated comment is a question
    1.2. checks if the iterated comment has been posted on tier 1 level
    1.3. checks if that comment is from the iAMA-Host himself or not

```

```
2. Now the posted question will be added to a global list, which will be used for csv writing and
chart generation
    later on
```

```
Args:
    id_of_thread (str) : Contains the id of the processed thread
    author_of_thread (str) : Contains the iAMA-Hosts name
Returns:
```

-

Definition at line 186 of file a_question_Tier_Distribution.py.

def a_question_Tier_Distribution.start_data_generation_for_analysis ()

```
Starts the data processing by swichting through the years
```

1. Triggers the data generation process and moves forward within the years
 - 1.1. By moving through the years a csv file will be created for every year
 - 1.2. Additionally an interactive chart will be plotted

```
Args:
-
Returns:
-
```

Definition at line 67 of file a_question_Tier_Distribution.py.

def a_question_Tier_Distribution.write_csv (*list_with_information*)

```
Creates a csv file containing all necessary information about the distribution of questions on the
tiers
```

```
This method iterates over the "current_year_question_list", which contains every single questions
of that year
and writes a csv file containing misc information about those questions.
```

```
One thing is to be said: The .csv file will be written in binary mode, therefore looking at them
in a plain text
editor could be a problem - please use excel for that.
I had to use "binary" mode, otherwise the questions-text could not be written into the csv file,
because windows
has some problem by converting some special chars to utf.
```

```
Args:
    list with information (list) : Contains information about questions for the current year
Returns:
-
```

Definition at line 345 of file a_question_Tier_Distribution.py.

Variable Documentation

int a_question_Tier_Distribution.argument_year_beginning = 0

Definition at line 454 of file a_question_Tier_Distribution.py.

int a_question_Tier_Distribution.argument_year_ending = 0

Definition at line 460 of file a_question_Tier_Distribution.py.

list a_question_Tier_Distribution.current_year_question_list = []

Definition at line 476 of file a_question_Tier_Distribution.py.

list a_question_Tier_Distribution.data_to_give_plotly = []

Definition at line 491 of file a_question_Tier_Distribution.py.

list a_question_Tier_Distribution.global_year_question_list = []

Definition at line 479 of file a_question_Tier_Distribution.py.

a_question_Tier_Distribution.mongo_DB_Client_Instance = None

Definition at line 464 of file a_question_Tier_Distribution.py.

a_question_Tier_Distribution.mongo_DB_Comments_Instance = None

Definition at line 473 of file a_question_Tier_Distribution.py.

a_question_Tier_Distribution.mongo_DB_Thread_Collection = None

Definition at line 470 of file a_question_Tier_Distribution.py.

a_question_Tier_Distribution.mongo_DB_Threads_Instance = None

Definition at line 467 of file a_question_Tier_Distribution.py.

int a_question_Tier_Distribution.year_actually_in_progress = 0

Definition at line 457 of file a_question_Tier_Distribution.py.

a_thread_Lifespan_N_Average_Commenttime Namespace Reference

Functions

- def [check_script_arguments](#) ()
- def [initialize_mongo_db_parameters](#) (actually_processed_year)
- def [start_data_generation_for_analysis](#) ()
- def [prepare_data_for_graph_life_span](#) ()
- def [prepare_data_for_comment_time](#) ()
- def [generate_data_to_be_analyzed](#) ()
- def [calculate_time_difference](#) (id_of_thread, creation_date_of_thread)
- def [write_csv](#) (list_with_information)
- def [add_thread_list_to_global_list](#) (list_to_append)
- def [prepare_dict_by_time_separation_for_comment_time](#) ()
- def [plot_generated_data](#) ()

Variables

- int [argument_year_beginning](#) = 0
- string [argument_calculation](#) = ""
- int [argument_year_ending](#) = 0
- int [year_actually_in_progress](#) = 0
- string [argument_plot_time_unit](#) = ""
- [mongo_DB_Client_Instance](#) = None
- [mongo_DB_Threads_Instance](#) = None
- [mongo_DB_Thread_Collection](#) = None
- [mongo_DB_Comments_Instance](#) = None
- list [global_thread_list](#) = []
- list [temp_time_difference_list](#) = []
- list [list_with_currents_year_infos](#) = []
- list [data_to_give_plotly](#) = []

Function Documentation

def a_thread_Lifespan_N_Average_Commenttime.add_thread_list_to_global_list (*list_to_append*)

```
Adds all elements of for the current year into a global list. This global list will be written into
a csv file
later on

1. This method simply checks whether both strings match each other or not.
   I have built this extra method to have a better overview in the main code..

Args:
    list_to_append (list) : The list which will be iterated over and which elements will be added
to the global list
Returns:
    -
```

Definition at line 742 of file a_thread_Lifespan_N_Average_Commenttime.py.

def a_thread_Lifespan_N_Average_Commenttime.calculate_time_difference (id_of_thread, creation_date_of_thread)

Calculates the difference between thread creation date and the last comment found in that thread

1. The creation date of a thread gets determined
2. Then the comments will be iterated over, creating a dictionary which is structured as follows:


```
{
    ('first_Comment_After_Thread_Started', int),
    ('thread_life_span', int),
    ('arithmetic Mean Response Time', int),
    ('median_Response_Time', int),
    ('id')
}
```
3. That returned dictionary will be appended to a global list
4. That List will be iterated later on and the appropriate graph will be plotted

Args:

id of thread (str) : The string which contains the id of the actually processed thread

creation_date_of_thread (str) : The string which contains the creation date of the thread (in epoch formatation)

Returns:

dict_to_be_returned (dict) : Containing information about the time difference

Definition at line 480 of file a_thread_Lifespan_N_Average_Commenttime.py.

def a_thread_Lifespan_N_Average_Commenttime.check_script_arguments ()

Checks if enough and correct arguments have been given to run this script adequate

1. It checks in the first instance if enough arguments have been given
2. Then necessary variables will be filled with appropriate values

Args:

-

Returns:

-

Definition at line 21 of file a_thread_Lifespan_N_Average_Commenttime.py.

def a_thread_Lifespan_N_Average_Commenttime.generate_data_to_be_analyzed ()

Generates the data which will be analyzed

1. This method iterates over every thread
 - 1.1. It filters if that iterated thread is an iAMA-request or not
 - 1.1.1. If yes: this thread gets skipped and the next one will be processed
 - 1.1.2. If no: this thread will be processed
2. If the thread gets processed it will receive the life span and other information about the thread as dictionary
3. This dictionary will be added to a global list and will be plotted later on

Args:

-

Returns:

-

Definition at line 422 of file a_thread_Lifespan_N_Average_Commenttime.py.

def a_thread_Lifespan_N_Average_Commenttime.initialize_mongo_db_parameters (actually_processed_year)

```
Instantiates all necessary variables for the correct usage of the mongoDB-Client

Args:
    actually_processed_year (int) : The year with which parameters the database should be accessed
Returns:
    -
```

Definition at line 48 of file a_thread_Lifespan_N_Average_Commenttime.py.

def a_thread_Lifespan_N_Average_Commenttime.plot_generated_data ()

```
Plots the data which is to be generated

1. This method plots the data which has been calculated before by using Pltoly-Framework within
a self written class

Args:
    -
Returns:
    -
```

Definition at line 881 of file a_thread_Lifespan_N_Average_Commenttime.py.

def a_thread_Lifespan_N_Average_Commenttime.prepare_data_for_comment_time ()

```
Prepares the average mean comment time per thread

Args:
    -
Returns:
    -
```

Definition at line 316 of file a_thread_Lifespan_N_Average_Commenttime.py.

def a_thread_Lifespan_N_Average_Commenttime.prepare_data_for_graph_life_span ()

```
Calculates the distribution of single values regarding the chosen time argument

Args:
    -
Returns:
    -
```

Definition at line 215 of file a_thread_Lifespan_N_Average_Commenttime.py.

def a_thread_Lifespan_N_Average_Commenttime.prepare_dict_by_time_separation_for_comment_time ()

```
Restructures the dictionary which is to be plotted for the display of the average mean comment time

1. This method processes the data in dependence of the committed time

Args:
    -
Returns:
    -
```

Definition at line 761 of file a_thread_Lifespan_N_Average_Commenttime.py.

def a_thread_Lifespan_N_Average_Commenttime.start_data_generation_for_analysis ()

```
Starts the data processing by swichting through the years

1. Triggers the data generation process and moves forward within the years
  1.1. By moving through the years a csv file will be created for every year
  1.2. Additionally an interactive chart will be plotted

Args:
-
Returns:
-
```

Definition at line 68 of file a_thread_Lifespan_N_Average_Commenttime.py.

def a_thread_Lifespan_N_Average_Commenttime.write_csv (*list_with_information*)

```
Creates a csv file containing all necessary information about the life span of a thread and various
information
  about comments

Args:
  list with information (list) : Contains various information about thread and comment time
Returns:
-
```

Definition at line 685 of file a_thread_Lifespan_N_Average_Commenttime.py.

Variable Documentation

string a_thread_Lifespan_N_Average_Commenttime.argument_calculation = ""

Definition at line 900 of file a_thread_Lifespan_N_Average_Commenttime.py.

string a_thread_Lifespan_N_Average_Commenttime.argument_plot_time_unit = ""

Definition at line 909 of file a_thread_Lifespan_N_Average_Commenttime.py.

int a_thread_Lifespan_N_Average_Commenttime.argument_year_beginning = 0

Definition at line 897 of file a_thread_Lifespan_N_Average_Commenttime.py.

int a_thread_Lifespan_N_Average_Commenttime.argument_year_ending = 0

Definition at line 903 of file a_thread_Lifespan_N_Average_Commenttime.py.

list a_thread_Lifespan_N_Average_Commenttime.data_to_give_plotly = []

Definition at line 945 of file a_thread_Lifespan_N_Average_Commenttime.py.

list a_thread_Lifespan_N_Average_Commenttime.global_thread_list = []

Definition at line 924 of file a_thread_Lifespan_N_Average_Commenttime.py.

list a_thread_Lifespan_N_Average_Commenttime.list_with_currents_year_infos = []

Definition at line 930 of file a_thread_Lifespan_N_Average_Commenttime.py.

a_thread_Lifespan_N_Average_Commenttime.mongo_DB_Client_Instance = None

Definition at line 912 of file a_thread_Lifespan_N_Average_Commenttime.py.

a_thread_Lifespan_N_Average_Commenttime.mongo_DB_Comments_Instance = None

Definition at line 921 of file a_thread_Lifespan_N_Average_Commenttime.py.

a_thread_Lifespan_N_Average_Commenttime.mongo_DB_Thread_Collection = None

Definition at line 918 of file a_thread_Lifespan_N_Average_Commenttime.py.

a_thread_Lifespan_N_Average_Commenttime.mongo_DB_Threads_Instance = None

Definition at line 915 of file a_thread_Lifespan_N_Average_Commenttime.py.

list a_thread_Lifespan_N_Average_Commenttime.temp_time_difference_list = []

Definition at line 927 of file a_thread_Lifespan_N_Average_Commenttime.py.

int a_thread_Lifespan_N_Average_Commenttime.year_actually_in_progress = 0

Definition at line 906 of file a_thread_Lifespan_N_Average_Commenttime.py.

a_thread_Lifespan_N_Average_Questiontime Namespace Reference

Functions

- def [check_script_arguments](#) ()
- def [initialize_mongo_db_parameters](#) (actually_processed_year)
- def [start_data_generation_for_analysis](#) ()
- def [prepare_data_for_graph_life_span](#) ()
- def [prepare_data_for_comment_time](#) ()
- def [generate_data_to_be_analyzed](#) ()
- def [calculate_time_difference](#) (id_of_thread, creation_date_of_thread)
- def [write_csv](#) (list_with_information)
- def [add_thread_list_to_global_list](#) (list_to_append)
- def [prepare_dict_by_time_separation_for_comment_time](#) ()
- def [plot_generated_data](#) ()

Variables

- int [argument_year_beginning](#) = 0
- string [argument_calculation](#) = ""
- int [argument_year_ending](#) = 0
- int [year_actually_in_progress](#) = 0
- string [argument_plot_time_unit](#) = ""
- [mongo_DB_Client_Instance](#) = None
- [mongo_DB_Threads_Instance](#) = None
- [mongo_DB_Thread_Collection](#) = None
- [mongo_DB_Comments_Instance](#) = None
- list [global_thread_list](#) = []
- list [temp_time_difference_list](#) = []
- list [list_with_currents_year_infos](#) = []
- list [data_to_give_plotly](#) = []

Function Documentation

def a_thread_Lifespan_N_Average_Questiontime.add_thread_list_to_global_list (*list_to_append*)

```
Adds all elements of for the current year into a global list. This global list will be written into
a csv file
later on

1. This method simply checks whether both strings match each other or not.
   I have built this extra method to have a better overview in the main code..

Args:
    list_to_append (list) : The list which will be iterated over and which elements will be added
to the global list
Returns:
    -
```

Definition at line 751 of file a_thread_Lifespan_N_Average_Questiontime.py.

**def a_thread_Lifespan_N_Average_Questiontime.calculate_time_difference (*id_of_thread*,
creation_date_of_thread)**

Calculates the difference between thread creation date and the last comment found in that thread

1. The creation date of a thread gets determined
2. Then the comments will be iterated over, creating a dictionary which is structured as follows:

```
{
    ('first_Comment_After_Thread_Started', int),
    ('thread_life_span', int),
    ('arithmetic_Mean_Response_Time', int),
    ('median Response Time', int),
    ('id')
}
```
3. That returned dictionary will be appended to a global list
4. That List will be iterated later on and the appropriate graph will be plotted

Args:

`id_of_thread (str)` : The string which contains the id of the actually processed thread
`creation date of thread (str)` : The string which contains the creation date of the thread (in epoch formatation)

Returns:

`dict_to_be_returned (dict)` : Containing information about the time difference

Definition at line 480 of file `a_thread_Lifespan_N_Average_Questiontime.py`.

def a_thread_Lifespan_N_Average_Questiontime.check_script_arguments ()

Checks if enough and correct arguments have been given to run this script adequate

1. It checks in the first instance if enough arguments have been given
2. Then necessary variables will be filled with appropriate values

Args:

-

Returns:

-

Definition at line 21 of file `a_thread_Lifespan_N_Average_Questiontime.py`.

def a_thread_Lifespan_N_Average_Questiontime.generate_data_to_be_analyzed ()

Generates the data which will be analyzed

1. This method iterates over every thread
 - 1.1. It filters if that iterated thread is an iAMA-request or not
 - 1.1.1. If yes: this thread gets skipped and the next one will be processed
 - 1.1.2. If no: this thread will be processed
2. If the thread gets processed it will receive the life span and other information about the thread as dictionary
3. This dictionary will be added to a global list and will be plotted later on

Args:

-

Returns:

-

Definition at line 422 of file `a_thread_Lifespan_N_Average_Questiontime.py`.

def a_thread_Lifespan_N_Average_Questiontime.initialize_mongo_db_parameters (*actually_processed_year*)

Instantiates all necessary variables for the correct usage of the mongoDB-Client

Args:

`actually_processed_year (int)` : The year with which parameters the database should be accessed

```
Returns:
-
```

Definition at line 48 of file a_thread_Lifespan_N_Average_Questiontime.py.

def a_thread_Lifespan_N_Average_Questiontime.plot_generated_data ()

```
Plots the data which is to be generated

1. This method plots the data which has been calculated before by using Pltoly-Framework within
a self written class

Args:
-
Returns:
-
```

Definition at line 890 of file a_thread_Lifespan_N_Average_Questiontime.py.

def a_thread_Lifespan_N_Average_Questiontime.prepare_data_for_comment_time ()

```
Prepares the average mean comment time per thread

Args:
-
Returns:
-
```

Definition at line 316 of file a_thread_Lifespan_N_Average_Questiontime.py.

def a_thread_Lifespan_N_Average_Questiontime.prepare_data_for_graph_life_span ()

```
Calculates the distribution of single values regarding the chosen time argument

Args:
-
Returns:
-
```

Definition at line 215 of file a_thread_Lifespan_N_Average_Questiontime.py.

def a_thread_Lifespan_N_Average_Questiontime.prepare_dict_by_time_separation_for_comment_time ()

```
Restructures the dictionary which is to be plotted for the display of the average mean comment time

1. This method processes the data in dependence of the committed time

Args:
-
Returns:
-
```

Definition at line 770 of file a_thread_Lifespan_N_Average_Questiontime.py.

def a_thread_Lifespan_N_Average_Questiontime.start_data_generation_for_analysis ()

```
Starts the data processing by swichting through the years

1. Triggers the data generation process and moves forward within the years
  1.1. By moving through the years a csv file will be created for every year
  1.2. Additionally an interactive chart will be plotted

Args:
-
Returns:
-
```

Definition at line 68 of file a_thread_Lifespan_N_Average_Questiontime.py.

def a_thread_Lifespan_N_Average_Questiontime.write_csv (*list_with_information*)

```
Creates a csv file containing all necessary information about the life span of a thread and various
information
  about comments

Args:
  list with information (list) : Contains various information about thread and comment time
Returns:
-
```

Definition at line 694 of file a_thread_Lifespan_N_Average_Questiontime.py.

Variable Documentation

string a_thread_Lifespan_N_Average_Questiontime.argument_calculation = ""

Definition at line 909 of file a_thread_Lifespan_N_Average_Questiontime.py.

string a_thread_Lifespan_N_Average_Questiontime.argument_plot_time_unit = ""

Definition at line 918 of file a_thread_Lifespan_N_Average_Questiontime.py.

int a_thread_Lifespan_N_Average_Questiontime.argument_year_beginning = 0

Definition at line 906 of file a_thread_Lifespan_N_Average_Questiontime.py.

int a_thread_Lifespan_N_Average_Questiontime.argument_year_ending = 0

Definition at line 912 of file a_thread_Lifespan_N_Average_Questiontime.py.

list a_thread_Lifespan_N_Average_Questiontime.data_to_give_plotly = []

Definition at line 954 of file a_thread_Lifespan_N_Average_Questiontime.py.

list a_thread_Lifespan_N_Average_Questiontime.global_thread_list = []

Definition at line 933 of file a_thread_Lifespan_N_Average_Questiontime.py.

list a_thread_Lifespan_N_Average_Questiontime.list_with_currents_year_infos = []

Definition at line 939 of file a_thread_Lifespan_N_Average_Questiontime.py.

a_thread_Lifespan_N_Average_Questiontime.mongo_DB_Client_Instance = None

Definition at line 921 of file a_thread_Lifespan_N_Average_Questiontime.py.

a_thread_Lifespan_N_Average_Questiontime.mongo_DB_Comments_Instance = None

Definition at line 930 of file a_thread_Lifespan_N_Average_Questiontime.py.

a_thread_Lifespan_N_Average_Questiontime.mongo_DB_Thread_Collection = None

Definition at line 927 of file a_thread_Lifespan_N_Average_Questiontime.py.

a_thread_Lifespan_N_Average_Questiontime.mongo_DB_Threads_Instance = None

Definition at line 924 of file a_thread_Lifespan_N_Average_Questiontime.py.

list a_thread_Lifespan_N_Average_Questiontime.temp_time_difference_list = []

Definition at line 936 of file a_thread_Lifespan_N_Average_Questiontime.py.

int a_thread_Lifespan_N_Average_Questiontime.year_actually_in_progress = 0

Definition at line 915 of file a_thread_Lifespan_N_Average_Questiontime.py.

ar_analyzer Namespace Reference

Functions

- def [average_means_of_values_f_threads](#) ()
- def [std_derivation](#) ()

Variables

- [thread_information](#)
- [thread_year](#) = [thread_information](#)['Year']
- [thread_id](#) = [thread_information](#)['Thread id']
- [thread_author](#) = [thread_information](#)['Thread author']
- [thread_ups](#) = [thread_information](#)['Thread ups']
- [thread_downs](#) = [thread_information](#)['Thread downs']
- [thread_creation_time_stamp](#) = [thread_information](#)['Thread creation time stamp']
- [thread_average_comment_vote_score_total](#)
- [thread_average_comment_vote_score_tier_1](#)
- [thread_average_comment_vote_score_tier_x](#)
- [thread_average_question_vote_score_total](#)
- [thread_average_question_vote_score_tier_1](#)
- [thread_average_question_vote_score_tier_x](#)
- [thread_num_comments_total_skewed](#)
- [thread_num_comments_total](#) = [thread_information](#)['Thread num comments total']
- [thread_num_comments_tier_1](#) = [thread_information](#)['Thread num comments tier 1']
- [thread_num_comments_tier_x](#) = [thread_information](#)['Thread num comments tier x']
- [thread_num_questions_total](#) = [thread_information](#)['Thread num questions total']
- [thread_num_questions_tier_1](#) = [thread_information](#)['Thread num questions tier 1']
- [thread_num_questions_tier_x](#) = [thread_information](#)['Thread num questions tier x']
- [thread_num_questions_answered_by_iam_a_host_total](#)
- [thread_num_questions_answered_by_iam_a_host_tier_1](#)
- [thread_num_questions_answered_by_iam_a_host_tier_x](#)
- [thread_num_comments_answered_by_iam_a_host_total](#)
- [thread_num_comments_answered_by_iam_a_host_tier_1](#)
- [thread_num_comments_answered_by_iam_a_host_tier_x](#)
- [thread_average_reaction_time_between_comments_total](#)
- [thread_average_reaction_time_between_comments_tier_1](#)
- [thread_average_reaction_time_between_comments_tier_x](#)
- [thread_average_reaction_time_between_questions_total](#)
- [thread_average_reaction_time_between_questions_tier_1](#)
- [thread_average_reaction_time_between_questions_tier_x](#)
- [thread_average_response_to_comment_time_iam_a_host_total](#)
- [thread_average_response_to_comment_time_iam_a_host_tier_1](#)
- [thread_average_response_to_comment_time_iam_a_host_tier_x](#)
- [thread_average_response_to_question_time_iam_a_host_total](#)
- [thread_average_response_to_question_time_iam_a_host_tier_1](#)
- [thread_average_response_to_question_time_iam_a_host_tier_x](#)
- [thread_amount_of_questioners_total](#)
- [thread_amount_of_questioners_tier_1](#)
- [thread_amount_of_questioners_tier_x](#)
- [thread_amount_of_commentators_total](#)
- [thread_amount_of_commentators_tier_1](#)
- [thread_amount_of_commentators_tier_x](#)
- [thread_life_span_until_last_comment](#)

- [thread life span until last question](#)
-

Function Documentation

def ar_analyzer.average_means_of_values_f_threads ()

```
Calculation of the average means of different values  
Args:  
-  
Returns:  
-
```

Definition at line 110 of file ar_analyzer.py.

def ar_analyzer.std_derivation ()

Definition at line 259 of file ar_analyzer.py.

Variable Documentation

ar_analyzer.thread_amount_of_commentators_tier_1

```
Initial value: 1 = thread_information[  
2 'Thread amount of commentators tier 1']
```

Definition at line 98 of file ar_analyzer.py.

ar_analyzer.thread_amount_of_commentators_tier_x

```
Initial value: 1 = thread_information[  
2 'Thread amount of commentators tier x']
```

Definition at line 100 of file ar_analyzer.py.

ar_analyzer.thread_amount_of_commentators_total

```
Initial value: 1 = thread_information[  
2 'Thread amount of commentators total']
```

Definition at line 96 of file ar_analyzer.py.

ar_analyzer.thread_amount_of_questioners_tier_1

```
Initial value: 1 = thread_information[  
2 'Thread amount of questioners tier 1']
```

Definition at line 91 of file ar_analyzer.py.

ar_analyzer.thread_amount_of_questioners_tier_x

```
Initial value: 1 = thread_information[  
2 'Thread amount of questioners tier x']
```

Definition at line 93 of file ar_analyzer.py.

ar_analyzer.thread_amount_of_questioners_total

```
Initial value: 1 = thread_information[  
2 'Thread amount of questioners total']
```

Definition at line 89 of file ar_analyzer.py.

ar_analyzer.thread_author = [thread_information](#)['Thread author']

Definition at line 17 of file ar_analyzer.py.

ar_analyzer.thread_average_comment_vote_score_tier_1

```
Initial value: 1 = thread_information[
2 'Thread average comment vote score tier 1']
```

Definition at line 25 of file ar_analyzer.py.

ar_analyzer.thread_average_comment_vote_score_tier_x

```
Initial value: 1 = thread_information[
2 'Thread average comment vote score tier x']
```

Definition at line 27 of file ar_analyzer.py.

ar_analyzer.thread_average_comment_vote_score_total

```
Initial value: 1 = thread_information[
2 'Thread average comment vote score total']
```

Definition at line 22 of file ar_analyzer.py.

ar_analyzer.thread_average_question_vote_score_tier_1

```
Initial value: 1 = thread_information[
2 'Thread average question vote score tier 1']
```

Definition at line 32 of file ar_analyzer.py.

ar_analyzer.thread_average_question_vote_score_tier_x

```
Initial value: 1 = thread_information[
2 'Thread average question vote score tier x']
```

Definition at line 34 of file ar_analyzer.py.

ar_analyzer.thread_average_question_vote_score_total

```
Initial value: 1 = thread_information[
2 'Thread average question vote score total']
```

Definition at line 30 of file ar_analyzer.py.

ar_analyzer.thread_average_reaction_time_between_comments_tier_1

```
Initial value: 1 = thread_information[
2 'Thread average reaction time between comments tier 1']
```

Definition at line 63 of file ar_analyzer.py.

ar_analyzer.thread_average_reaction_time_between_comments_tier_x

```
Initial value: 1 = thread_information[
2 'Thread average reaction time between comments tier x']
```

Definition at line 65 of file ar_analyzer.py.

ar_analyzer.thread_average_reaction_time_between_comments_total

```
Initial value: 1 = thread_information[
2 'Thread average reaction time between comments total']
```

Definition at line 61 of file ar_analyzer.py.

ar_analyzer.thread_average_reaction_time_between_questions_tier_1

```
Initial value: 1 = thread_information[
2 'Thread average reaction time between questions tier 1']
```

Definition at line 70 of file ar_analyzer.py.

ar_analyzer.thread_average_reaction_time_between_questions_tier_x

```
Initial value: 1 = thread_information[
                2 'Thread average reaction time between questions tier x']
```

Definition at line 72 of file ar_analyzer.py.

ar_analyzer.thread_average_reaction_time_between_questions_total

```
Initial value: 1 = thread_information[
                2 'Thread average reaction time between questions total']
```

Definition at line 68 of file ar_analyzer.py.

ar_analyzer.thread_average_response_to_comment_time_iama_host_tier_1

```
Initial value: 1 = thread_information[
                2 'Thread average response to comment time iama host tier 1']
```

Definition at line 77 of file ar_analyzer.py.

ar_analyzer.thread_average_response_to_comment_time_iama_host_tier_x

```
Initial value: 1 = thread_information[
                2 'Thread average response to comment time iama host tier x']
```

Definition at line 79 of file ar_analyzer.py.

ar_analyzer.thread_average_response_to_comment_time_iama_host_total

```
Initial value: 1 = thread_information[
                2 'Thread average response to comment time iama host total']
```

Definition at line 75 of file ar_analyzer.py.

ar_analyzer.thread_average_response_to_question_time_iama_host_tier_1

```
Initial value: 1 = thread_information[
                2 'Thread average response to question time iama host tier 1']
```

Definition at line 84 of file ar_analyzer.py.

ar_analyzer.thread_average_response_to_question_time_iama_host_tier_x

```
Initial value: 1 = thread_information[
                2 'Thread average response to question time iama host tier x']
```

Definition at line 86 of file ar_analyzer.py.

ar_analyzer.thread_average_response_to_question_time_iama_host_total

```
Initial value: 1 = thread_information[
                2 'Thread average response to question time iama host total']
```

Definition at line 82 of file ar_analyzer.py.

ar_analyzer.thread_creation_time_stamp = [thread_information](#)['Thread creation time stamp']

Definition at line 20 of file ar_analyzer.py.

ar_analyzer.thread_downs = [thread_information](#)['Thread downs']

Definition at line 19 of file ar_analyzer.py.

ar_analyzer.thread_id = [thread_information](#)['Thread id']

Definition at line 16 of file ar_analyzer.py.

ar_analyzer.thread_information

```
Initial value: 1 = pandas.read_csv(  
2     'd_create_Big_CSV_2009_until_2016_BIGDATA_ALL.csv',  
3     sep=',',  
4     na_values="None")
```

Definition at line 10 of file ar_analyzer.py.

ar_analyzer.thread_life_span_until_last_comment

```
Initial value: 1 = thread_information[  
2     'Thread life span until last comment']
```

Definition at line 104 of file ar_analyzer.py.

ar_analyzer.thread_life_span_until_last_question

```
Initial value: 1 = thread_information[  
2     'Thread life span until last question']
```

Definition at line 106 of file ar_analyzer.py.

ar_analyzer.thread_num_comments_answered_by_iam_a_host_tier_1

```
Initial value: 1 = thread_information[  
2     'Thread num comments answered by iama host tier 1']
```

Definition at line 56 of file ar_analyzer.py.

ar_analyzer.thread_num_comments_answered_by_iam_a_host_tier_x

```
Initial value: 1 = thread_information[  
2     'Thread num comments answered by iama host tier x']
```

Definition at line 58 of file ar_analyzer.py.

ar_analyzer.thread_num_comments_answered_by_iam_a_host_total

```
Initial value: 1 = thread_information[  
2     'Thread num comments answered by iama host total']
```

Definition at line 54 of file ar_analyzer.py.

ar_analyzer.thread_num_comments_tier_1 = [thread_information](#)['Thread num comments tier 1']

Definition at line 40 of file ar_analyzer.py.

ar_analyzer.thread_num_comments_tier_x = [thread_information](#)['Thread num comments tier x']

Definition at line 41 of file ar_analyzer.py.

ar_analyzer.thread_num_comments_total = [thread_information](#)['Thread num comments total']

Definition at line 39 of file ar_analyzer.py.

ar_analyzer.thread_num_comments_total_skewed

```
Initial value: 1 = thread_information[  
2     'Thread num comments total skewed']
```

Definition at line 37 of file ar_analyzer.py.

ar_analyzer.thread_num_questions_answered_by_iam_a_host_tier_1

```
Initial value: 1 = thread_information[  
2     'Thread num questions answered by iama host tier 1']
```

Definition at line 49 of file ar_analyzer.py.

ar_analyzer.thread_num_questions_answered_by_iama_host_tier_x

```
Initial value: 1 = thread_information[
                2      'Thread num questions answered by iama host tier x']
```

Definition at line 51 of file ar_analyzer.py.

ar_analyzer.thread_num_questions_answered_by_iama_host_total

```
Initial value: 1 = thread_information[
                2      'Thread num questions answered by iama host total']
```

Definition at line 47 of file ar_analyzer.py.

ar_analyzer.thread_num_questions_tier_1 = [thread_information](#)['Thread num questions tier 1']

Definition at line 44 of file ar_analyzer.py.

ar_analyzer.thread_num_questions_tier_x = [thread_information](#)['Thread num questions tier x']

Definition at line 45 of file ar_analyzer.py.

ar_analyzer.thread_num_questions_total = [thread_information](#)['Thread num questions total']

Definition at line 43 of file ar_analyzer.py.

ar_analyzer.thread_ups = [thread_information](#)['Thread ups']

Definition at line 18 of file ar_analyzer.py.

ar_analyzer.thread_year = [thread_information](#)['Year']

Definition at line 15 of file ar_analyzer.py.

c_crawl_Author_Information Namespace Reference

Functions

- def [check_script_arguments](#) ()
- def [initialize_mongo_db_parameters](#) (actually_processed_year)
- def [start_data_generation_for_analysis](#) ()
- def [generate_data_now](#) ()
- def [calculate_time_difference](#) (time_value_1, time_value_2)
- def [get_author_information](#) (name_of_author)

Variables

- int [argument_year_beginning](#) = 0
- int [year_actually_in_progress](#) = 0
- int [argument_year_ending](#) = 0
- string [argument_inverse_crawling](#) = ""
- [mongo_db_client_instance](#) = None
- [mongo_db_threads_instance](#) = None
- [mongo_db_thread_collection](#) = None
- [mongo_db_author_instance](#) = None
- [reddit_instance](#) = praw.Reddit(user_agent="University_Regensburg_iAMA_Crawler_0.001")

Function Documentation

def c_crawl_Author_Information.calculate_time_difference (*time_value_1*, *time_value_2*)

Calculates the time difference between two floats in epoch style and returns seconds

Args:

time_value_1 (float): The first time value to be used for calculation
time_value_2 (float): The second time value to be used for calculation

Returns:

time_diff_seconds (int): The amount of time difference in seconds

Definition at line 233 of file c_crawl_Author_Information.py.

def c_crawl_Author_Information.check_script_arguments ()

Checks if enough and correct arguments have been given to run this script adequate

1. It checks in the first instance if enough arguments have been given
2. Then necessary variables will be filled with appropriate values

Args:

-

Returns:

-

Definition at line 18 of file c_crawl_Author_Information.py.

def c_crawl_Author_Information.generate_data_now ()

```

Crawls author information and writes them into the mongoDB database with the name
'iAMA_Reddit_Authors'

It does this by first checking the given crawling direction. The ability to crawl bidirectional
allows you to build
up you database in a much more faster way, because you can start one instance crawling forward while
the other
instance crawls backward.

This method works in the following way:

1. Checks for crawling direction
2. It checks whether an iterated collection is no "system.indexes".
3. By iterating over all collections it checks for iAMA-Requests and skips them. Because we do not
want requests
in our dataset, because we want data of actually created iama threads

4. Now it will be checked whether the author already exists within the database (collection name).
This will be
done by always re-initialising the collection.names() which is necessary to always have a
up2date-overview!

    4.1. Whenever the author does not exist yet get the necessary information and write it into
the database

    4.2. Whenever the author does already exist skip that calculation part

Args:
-
Returns:
-

```

Definition at line 103 of file `c_crawl_Author_Information.py`.

def c_crawl_Author_Information.get_author_information (*name_of_author*)

```

Calculates various information about the author
Because I have created this script shortly before my evaluation everything listed here is not
outsourced by
written in a very sequential / procedural way, therefore I ask for you understanding.

The method does the following:

1. Referencing a reddit author object, which is necessary to get all that necessary data
2. Declaration of necessary variables for later assignment
3. Trial of receiving the authors birthday
    We have to try this here, because, if the account has already been deleted a http error
will be thrown
    and we would have to recrawl all that data.
4. Receival authors comment / link karma - amount
5. Trial of receiving of all links / comments the author every made
    Because it could happen, that there is an internal error in reddit ongoing (Error 500) which
will reset
    the connection and therefore we would have to recrawl all of our data
6. Iteration of all comments / links and therefore saving the time difference (in seconds)
between each created
    comment / link
7. Calculation of time difference between acc birth & first iama in seconds
8. Patching up a big dictionary which will be sorted (alphabetically correct)
9. Return that dictionary

Args:
    name_of_author (str): The name of the author which information need to be calculated
Returns:
    dict_to_be_returned (dict): Dictionary containing various information about the author. It will
be written

```

Definition at line 269 of file `c_crawl_Author_Information.py`.

def c_crawl_Author_Information.initialize_mongo_db_parameters (*actually_processed_year*)

```
Instantiates all necessary variables for the correct usage of the mongoDB client

Args:
    actually_processed_year (int) : The year with which parameters the database should be accessed
Returns:
    -
```

Definition at line 48 of file c_crawl_Author_Information.py.

def c_crawl_Author_Information.start_data_generation_for_analysis ()

```
Starts the data processing by swichting through the years
    After every year cycle the mongo db parameters will be reinitialized

Args:
    -
Returns:
    -
```

Definition at line 68 of file c_crawl_Author_Information.py.

Variable Documentation

string c_crawl_Author_Information.argument_inverse_crawling = ""

Definition at line 499 of file c_crawl_Author_Information.py.

int c_crawl_Author_Information.argument_year_beginning = 0

Definition at line 490 of file c_crawl_Author_Information.py.

int c_crawl_Author_Information.argument_year_ending = 0

Definition at line 496 of file c_crawl_Author_Information.py.

c_crawl_Author_Information.mongo_db_author_instance = None

Definition at line 511 of file c_crawl_Author_Information.py.

c_crawl_Author_Information.mongo_db_client_instance = None

Definition at line 502 of file c_crawl_Author_Information.py.

c_crawl_Author_Information.mongo_db_thread_collection = None

Definition at line 508 of file c_crawl_Author_Information.py.

c_crawl_Author_Information.mongo_db_threads_instance = None

Definition at line 505 of file c_crawl_Author_Information.py.

**c_crawl_Author_Information.reddit_instance =
praw.Reddit(user_agent="University_Regensburg_iAMA_Crawler_0.001")**

Definition at line 514 of file c_crawl_Author_Information.py.

int c_crawl_Author_Information.year_actually_in_progress = 0

Definition at line 493 of file c_crawl_Author_Information.py.

c_crawl_Differences Namespace Reference

Functions

- def [check_script_arguments](#) ()
- def [initialize_mongo_db_parameters](#) ()
- def [crawl_missing_collection_into_comments_database](#) (name_of_missing_collection)
- def [check_if_collection_is_missing_in_comments_database](#) ()
- def [crawl_missing_collection_into_threads_database](#) (name_of_missing_collection)
- def [check_if_collection_is_missing_in_threads_database](#) ()
- def [start_crawling_for_diffs](#) ()

Variables

- [mongo_DB_Client_Instance](#) = None
- [mongo_DB_Threads_Instance](#) = None
- [mongo_DB_Thread_Collection](#) = None
- [mongo_DB_Comments_Instance](#) = None
- [mongo_DB_Comments_Collection](#) = None
- string [argument_year_beginning](#) = ""
- string [argument_year_ending](#) = ""
- string [argument_inverse_crawling](#) = ""

Function Documentation

def c_crawl_Differences.check_if_collection_is_missing_in_comments_database ()

```
Checks if a specific collection (thread) is missing in the appropriate comments database

    The method starts the diff checking for all collections within the threads database.
    Whenever a thread exists in the comment database but not in the threads database it will be
    crawled from the
    reddit servers and written into the database.

Args:
    -
Returns:
    -
```

Definition at line 213 of file c_crawl_Differences.py.

def c_crawl_Differences.check_if_collection_is_missing_in_threads_database ()

```
Checks if a specific collection (thread) is missing in the appropriate threads database

    The method starts the diff checking for all collections within the threads database.
    Whenever a thread exists in the comment database but not in the threads database it will be
    crawled from the
    reddit servers and written into the database.

Args:
    -
Returns:
    -
```

Definition at line 353 of file c_crawl_Differences.py.

def c_crawl_Differences.check_script_arguments ()

```
Checks if enough and correct arguments have been given to run this script adequate

1. It checks in the first instance if enough arguments have been given
2. Then necessary variables will be filled with appropriate values

Args:
-
Returns:
-
```

Definition at line 14 of file c_crawl_Differences.py.

def c_crawl_Differences.crawl_missing_collection_into_comments_database (*name_of_missing_collection*)

```
Crawls a specific thread, which is missing in the comments database and writes the appropriate entry  
in the db

    The method works as follows:
    1. It checks whether that thread / collection is really missing (even when that has been done  
before, we check  
        it again here, just to make sure that collection has not been created in the meanwhile by  
another crawling  
        process.
    2. Now the comments will be crawled from the reddit servers with flattened hierarchy
    3. Yet the comments will be written into the appropriate comments database. The correct database  
will be  
        deviated from the threads creation timestamp.

Args:
    name_of_missing_collection (str) : The id of the collection which is actually missing in the  
comments database
Returns:
-
```

Definition at line 76 of file c_crawl_Differences.py.

def c_crawl_Differences.crawl_missing_collection_into_threads_database (*name_of_missing_collection*)

```
Crawls a specific thread, which is missing in the thread database and writes the appropriate entry  
in the db

    The method works as follows:
    1. It checks whether that thread / collection is really missing (even when that has been done  
before, we check  
        it again here, just to make sure that collection has not been created in the meanwhile by  
another crawling  
        process.
    2. Now the the thread will be crawled from the reddit servers
    3. Yet the thread will be written into the appropriate threads database. The correct database  
will be  
        deviated from the threads creation timestamp.

Args:
    name of missing collection (str) : The id of the collection which is actually missing in the  
comments database
Returns:
```

-

Definition at line 269 of file c_crawl_Differences.py.

def c_crawl_Differences.initialize_mongo_db_parameters ()

```
Instantiates all necessary variables for the correct usage of the mongoDB-Client  
Args:  
-  
Returns:  
-
```

Definition at line 49 of file c_crawl_Differences.py.

def c_crawl_Differences.start_crawling_for_diffs ()

```
This method starts the crawling, with the method you have defined in your arguments  
Args:  
-  
Returns:  
-
```

Definition at line 406 of file c_crawl_Differences.py.

Variable Documentation

string c_crawl_Differences.argument_inverse_crawling = ""

Definition at line 481 of file c_crawl_Differences.py.

string c_crawl_Differences.argument_year_beginning = ""

Definition at line 475 of file c_crawl_Differences.py.

string c_crawl_Differences.argument_year_ending = ""

Definition at line 478 of file c_crawl_Differences.py.

c_crawl_Differences.mongo_DB_Client_Instance = None

Definition at line 460 of file c_crawl_Differences.py.

c_crawl_Differences.mongo_DB_Comments_Collection = None

Definition at line 472 of file c_crawl_Differences.py.

c_crawl_Differences.mongo_DB_Comments_Instance = None

Definition at line 469 of file c_crawl_Differences.py.

c_crawl_Differences.mongo_DB_Thread_Collection = None

Definition at line 466 of file c_crawl_Differences.py.

c_crawl_Differences.mongo_DB_Threads_Instance = None

Definition at line 463 of file c_crawl_Differences.py.

c_crawl_Random_Author_Information Namespace Reference

Functions

- def [check_script_arguments](#) ()
- def [initialize_mongo_db_parameters](#) ()
- def [start_data_generation_for_analysis](#) ()
- def [generate_data_now](#) (randomized_author_name)
- def [calculate_time_difference](#) (time_value_1, time_value_2)
- def [get_author_information](#) (name_of_author)

Variables

- [argument_limit_crawling_amount](#) = None
- [mongo_db_client_instance](#) = None
- [mongo_db_random_author_instance](#) = None
- [mongo_db_random_author_collection](#) = None
- [mongo_db_iama_author_instance](#) = None
- [mongo_db_iama_author_collection](#) = None
- int [mongo_db_iama_author_collection_amount](#) = 0
- [reddit_instance](#) = praw.Reddit(user_agent="University_Regensburg_iAMA_Crawler_0.001")

Function Documentation

def c_crawl_Random_Author_Information.calculate_time_difference (*time_value_1*, *time_value_2*)

```
Calculates the time difference between two floats in epoch style and returns seconds

Args:
    time_value_1 (float): The first time value to be used for calculation
    time_value_2 (float): The second time value to be used for calculation
Returns:
    time_diff_seconds (int): The amount of time difference in seconds
```

Definition at line 159 of file c_crawl_Random_Author_Information.py.

def c_crawl_Random_Author_Information.check_script_arguments ()

```
Checks if enough and correct arguments have been given to run this script adequate

1. It checks in the first instance if enough arguments have been given
2. Then necessary variables will be filled with appropriate values

Args:
    -
Returns:
    -
```

Definition at line 17 of file c_crawl_Random_Author_Information.py.

def c_crawl_Random_Author_Information.generate_data_now (*randomized_author_name*)

```

Crawls author information and writes them into the mongoDB database with the name
'iAMA_Reddit_Authors_Random'

It does this by first checking the given crawling direction. The ability to crawl bidirectional
allows you to build
up you database in a much more faster way, because you can start one instance crawling forward while
the other
instance crawls backward.

This method works in the following way:

1. Checks for crawling direction
2. It checks whether an iterated collection is no "system.indexes".
3. By iterating over all collections it checks for iAMA-Requests and skips them. Because we do not
want requests
in our dataset, because we want data of actually created iama threads

4. Now it will be checked whether the author already exists within the database (collection name).
This will be
done by always re-initialising the collection.names() which is necessary to always have a
up2date-overview!

    4.1. Whenever the author does not exist yet get the necessary information and write it into
the database

    4.2. Whenever the author does already exist skip that calculation part

Args:
-
Returns:
-

```

Definition at line 107 of file `c_crawl_Random_Author_Information.py`.

`def c_crawl_Random_Author_Information.get_author_information (name_of_author)`

```

Calculates various information about the author
Because I have created this script shortly before my evaluation everything listed here is not
outsourced by
written in a very sequential / procedural way, therefore I ask for you understanding.

The method does the following:

1. Referencing a reddit author object, which is necessary to get all that necessary data
2. Declaration of necessary variables for later assignment
3. Trial of receiving the authors birthday
    We have to try this here, because, if the account has already been deleted a http error
will be thrown
    and we would have to recrawl all that data.
4. Receival authors comment / link karma - amount
5. Trial of receiving of all links / comments the author every made
    Because it could happen, that there is an internal error in reddit ongoing (Error 500) which
will reset
    the connection and therefore we would have to recrawl all of our data
6. Iteration of all comments / links and therefore saving the time difference (in seconds)
between each created
    comment / link
7. Calculation of time difference between acc birth & first iama in seconds
8. Patching up a big dictionary which will be sorted (alphabetically correct)
9. Return that dictionary

Args:
    name_of_author (str): The name of the author which information need to be calculated
Returns:
    dict_to_be_returned (dict): Dictionary containing various information about the author. It will
be written

```

Definition at line 195 of file `c_crawl_Random_Author_Information.py`.

def c_crawl_Random_Author_Information.initialize_mongo_db_parameters ()

```
Instantiates all necessary variables for the correct usage of the mongoDB client  
  
Args:  
-  
Returns:  
-
```

Definition at line 43 of file c_crawl_Random_Author_Information.py.

def c_crawl_Random_Author_Information.start_data_generation_for_analysis ()

```
Starts the data processing by swichting through the years  
After every year cycle the mongo db parameters will be reinitialized  
  
Args:  
-  
Returns:  
-
```

Definition at line 68 of file c_crawl_Random_Author_Information.py.

Variable Documentation

c_crawl_Random_Author_Information.argument_limit_crawling_amount = None

Definition at line 422 of file c_crawl_Random_Author_Information.py.

c_crawl_Random_Author_Information.mongo_db_client_instance = None

Definition at line 425 of file c_crawl_Random_Author_Information.py.

c_crawl_Random_Author_Information.mongo_db_iama_author_collection = None

Definition at line 437 of file c_crawl_Random_Author_Information.py.

int c_crawl_Random_Author_Information.mongo_db_iama_author_collection_amount = 0

Definition at line 440 of file c_crawl_Random_Author_Information.py.

c_crawl_Random_Author_Information.mongo_db_iama_author_instance = None

Definition at line 434 of file c_crawl_Random_Author_Information.py.

c_crawl_Random_Author_Information.mongo_db_random_author_collection = None

Definition at line 431 of file c_crawl_Random_Author_Information.py.

```
c_crawl_Random_Author_Information.mongo_db_random_author_instance = None
```

Definition at line 428 of file c_crawl_Random_Author_Information.py.

```
c_crawl_Random_Author_Information.reddit_instance =  
praw.Reddit(user_agent="University_Regensburg_iAMA_Crawler_0.001")
```

Definition at line 443 of file c_crawl_Random_Author_Information.py.

c_crawl_Threads_N_Comments Namespace Reference

Functions

- def [initialize_mongo_db_parameters](#) ()
- def [check_script_arguments](#) ()
- def [convert_argument_year_to_epoch](#) (year)
- def [crawl_data](#) ()
- def [crawl_threads](#) ()
- def [crawl_comments](#) ()
- def [check_if_coll_in_db_already_exists_up2date](#) (submission)

Variables

- [mongo_DB_Client_Instance](#) = None
- [reddit_Instance](#) = None
- [argument_crawl_type](#) = None
- [argument_year_beginning](#) = None
- [argument_year_end](#) = None
- [argument_hours_to_shift](#) = None
- [time_shift_difference](#)

Function Documentation

def c_crawl_Threads_N_Comments.check_if_coll_in_db_already_exists_up2date (*submission*)

Checks if a collection already exists in the database or not

This is necessary, otherwise thread information would be written into the database twice.
It works the following way:

1. Define a tolerance factor (necessary because reddit skews information about the amount of "upvotes"). Without defining that tolerance factor every thread would be created anew.
After messing around a few days I found this one to be the best value to work with
2. Create values for temporary values for checking
3. Check and recreate collection if necessary
4. Return appropriate boolean value if collection already existed within the database or not

Args:

 submission (Submission) : The thread which will be processed / iterated over at the moment

Returns:

 True / False (bool) : Whenever the collection already exists within the database (True) or not (False)

Definition at line 373 of file c_crawl_Threads_N_Comments.py.

def c_crawl_Threads_N_Comments.check_script_arguments ()

Checks if enough and correct arguments have been given to run this script adequate

1. It checks in the first instance if enough arguments have been given
2. Then necessary variables will be filled with appropriate values

```
Args:
-
Returns:
-
```

Definition at line 36 of file c_crawl_Threads_N_Comments.py.

def c_crawl_Threads_N_Comments.convert_argument_year_to_epoch (year)

```
"Converts" a given string into the appropriate epoch string format (int)

Args:
    year (str) : The year which will be "converted" into epoch format (necessary for correct PRAW
API behaviour)
Returns:
    year (int) : The year "converted" into epoch format as integer
```

Definition at line 71 of file c_crawl_Threads_N_Comments.py.

def c_crawl_Threads_N_Comments.crawl_comments ()

```
Crawls thread information and writes them into the mongoDB storage
It works as following:

1. At first an attempt to the amazon cloud search will be made, with necessary parameters which
returns an object,
    of the class "Generator" which contains all comments for the given / crawled time windows

2. After that the "Generator"s elements will be iterated over

    2.1. It will be checked if that iterated collection already exists within the database or not

        2.2.1. If it already exists, it will be checked whether if it is up to date or not
            2.2.1.1. If up2date: do nothing
            2.2.1.2. If not up2date: drop that collection within the database and crawl the
collection anew

        2.2.2. If it does not yet exist: create that collection in the database with the necessary
information

3. Whenever there are no elements left to iterate over the time crawling window will be shifted
into the future by
    using the given amount in hours (fourth argument), whenever the ending year (third argument)
is not reached yet

Args:
-
Returns:
-
```

Definition at line 258 of file c_crawl_Threads_N_Comments.py.

def c_crawl_Threads_N_Comments.crawl_data ()

```
Crawls data from reddit, depending on the first argument (threads / comments) you give the script

Args:
-
Returns:
-
```

Definition at line 121 of file c_crawl_Threads_N_Comments.py.

def c_crawl_Threads_N_Comments.crawl_threads ()

```
Crawls thread information and writes them into the mongoDB storage
It works as follwoing:

1. At first an attempt to the amazon cloud search will be made, with necessary parameters which
   returns an object,
   of the class "Generator" which contains all threads for the given / crawled time windows

2. After that the "Generator"s elements will be iterated over

   2.1. It will be checked if that iterated collection already exists within the database or not

       2.2.1. If it already exists, it will be checked whether if it is up to date or not
       2.2.1.1. If up2date: do nothing
       2.2.1.2. If not up2date: drop that collection within the database and crawl the
       collection anew

       2.2.2. If it does not yet exist: create that collection in the database with the necessary
       information

3. Whenever there are no elements left to iterate over the time crawling window will be shifted
   into the future by
   using the given amount in hours (third argument), whenever the ending year (second argument)
   is not reached yet

Args:
-
Returns:
-
```

Definition at line 140 of file c_crawl_Threads_N_Comments.py.

def c_crawl_Threads_N_Comments.initialize_mongo_db_parameters ()

```
Instantiates all necessary variables for the correct usage of the mongoDB-Client

Args:
-
Returns:
-
```

Definition at line 21 of file c_crawl_Threads_N_Comments.py.

Variable Documentation

c_crawl_Threads_N_Comments.argument_crawl_type = None

Definition at line 480 of file c_crawl_Threads_N_Comments.py.

c_crawl_Threads_N_Comments.argument_hours_to_shift = None

Definition at line 496 of file c_crawl_Threads_N_Comments.py.

c_crawl_Threads_N_Comments.argument_year_beginning = None

Definition at line 483 of file c_crawl_Threads_N_Comments.py.

c_crawl_Threads_N_Comments.argument_year_end = None

Definition at line 486 of file c_crawl_Threads_N_Comments.py.

c_crawl_Threads_N_Comments.mongo_DB_Client_Instance = None

Definition at line 474 of file c_crawl_Threads_N_Comments.py.

c_crawl_Threads_N_Comments.reddit_Instance = None

Definition at line 477 of file c_crawl_Threads_N_Comments.py.

c_crawl_Threads_N_Comments.time_shift_difference

```
Initial value: 1 = int(  
2     round(time.mktime(  
3         (datetime.fromtimestamp(argument_year_beginning) +  
4             timedelta(  
5                 hours=argument_hours_to_shift)  
6             ).timetuple()  
7     )  
8     )  
9 )
```

Definition at line 516 of file c_crawl_Threads_N_Comments.py.

d_create_Big_CSV_ar Namespace Reference

Functions

- def [check_script_arguments](#) ()
- def [initialize_mongo_db_parameters](#) (actually_processed_year)
- def [start_data_generation_for_analysis](#) ()
- def [generate_data](#) ()
- def [process_specific_thread](#) (thread_id, thread_creation_time_stamp, thread_author)
- def [check_if_comment_is_a_question](#) (given_string)
- def [check_if_comment_is_on_tier_1](#) (comment_parent_id)
- def [check_if_comment_is_not_from_thread_author](#) (author_of_thread, comment_author)
- def [check_if_comment_has_been_answered_by_thread_author](#) (author_of_thread, comment_actual_id, comments_cursor)
- def [calculate_time_difference](#) (comment_time_stamp, answer_time_stamp_iama_host)
- def [calculate_reaction_time_average](#) (list_to_be_processed, thread_creation_time_stamp)
- def [calculate_life_span](#) (thread_creation_time_stamp, time_value_of_last_comment, time_value_of_last_question)
- def [add_actual_year_list_to_global_list](#) (list_to_append)
- def [write_csv_data](#) (list_with_information)

Variables

- int [argument_year_beginning](#) = 0
- int [argument_year_ending](#) = 0
- int [year_actually_in_progress](#) = 0
- list [list_current_year](#) = []
- list [list_global_year](#) = []

Function Documentation

def d_create_Big_CSV_ar.add_actual_year_list_to_global_list (*list_to_append*)

Iterates over a given list with thread information and adds every single element to a global list
The global list will be printed to csv in the end

Args:

list_to_append (list) : List with thread information which will be appended to a global list

Returns:

-

Definition at line 1028 of file d_create_Big_CSV_ar.py.

def d_create_Big_CSV_ar.calculate_life_span (*thread_creation_time_stamp*, *time_value_of_last_comment*, *time_value_of_last_question*)

Calculates the life span between to time stamps

1. The creation date of a thread gets determined
2. Then the comments will be iterated over, creating a dictionary which is structured as follows:
{
 ('first_Comment_After_Thread_Started', int),
 ('thread_life_span', int),
 ('arithmetic_Mean_Response_Time', int),

```

        ('median_Response_Time', int),
        ('id')
    }
3. That returned dictionary will be appended to a global list
4. That List will be iterated later on and the appropriate graph will be plotted

Args:
    thread_creation_time_stamp (float) : The time stamp (utc epoch) of the thread creation
    time value of last comment (float) : The time stamp (utc epoch) of the threads last comment
    time_value_of_last_question (float) : The time stamp (utc epoch) of the threads last question
Returns:
    dict_to_be_returned (dict) : Containing information about the time differences:
        Thread creation timestamp <-> Last question time stamp

```

Thread creation timestamp <-> Last comment time stamp

Definition at line 973 of file d_create_Big_CSV_ar.py.

**def d_create_Big_CSV_ar.calculate_reaction_time_average (*list_to_be_processed*,
thread_creation_time_stamp)**

```

Calculates the reaction time of a list with time values in it

Args:
    list to be processed (list) : The list which contains time values (utc epoch)
    thread_creation_time_stamp (str) : The string which contains the creation date of the thread
    (utc epoch)
Returns:
    None : Whenever there were no time values given

```

np.mean(time_difference) (float) : Time arithmetic mean of the reaction time in seconds

Definition at line 889 of file d_create_Big_CSV_ar.py.

**def d_create_Big_CSV_ar.calculate_time_difference (*comment_time_stamp*,
answer_time_stamp_iama_host)**

```

Calculates the time difference in seconds between the a comment and its answer from the iama host

1. The time stamps will be converted from epoch
    into float and afterwards into str again
    (necessary for correct subtraction)
2. Then the time stamps will be subtracted from each other
3. The containing time difference will be converted into seconds (int)

Args:
    comment_time_stamp (str): The time stamp of the comment
    answer_time_stamp_iama_host (str): The time stamp of the iAMA hosts answer
Returns:
    time_difference_in_seconds (int) : The time difference of the comment and its answer by the
    iAMA host in seconds

```

Definition at line 850 of file d_create_Big_CSV_ar.py.

def d_create_Big_CSV_ar.check_if_comment_has_been_answered_by_thread_author (
***author_of_thread*, *comment_actual_id*, *comments_cursor*)**

```

Checks whether both strings are equal or not

1. A dictionary containing flags whether that a question is answered by the host with the appropriate
    timestamp will
    be created in the beginning.
2. Then the method iterates over every comment within that thread
    1.1. Whenever an answer is from the iAMA hosts and the processed comments 'parent_id' matches
    the iAMA hosts

```

comments (answers) id, the returned dict will contain appropriate values and will be returned

1.2. If this is not the case, it will be returned in its default condition

Note: We take a list as 'comments cursor' and not a real cursor, because real cursors can be exhausted, which

could lead to, that not all comments will be iterated.. This is especially critical when you have to do

many iterations with only one cursor... [took me 8 hours to figure this "bug" out...]

Args:

author_of_thread (str) : The name of the thread author (iAMA-Host)

comment_actual_id (str) : The id of the actually processed comment

comments_cursor (list) : The list containing all comments

Returns:

True (bool): Whenever the strings do not match

False (bool): Whenever the strings do match

answered that given question)

Definition at line 802 of file d_create_Big_CSV_ar.py.

def d_create_Big_CSV_ar.check_if_comment_is_a_question (*given_string*)

Simply checks whether a given string is a question or not

This method simply checks whether a question mark exists within that string or not..

This is just that simple because messing around with natural processing kits to determine the semantic sense

would blow up my bachelor work...

Args:

given_string (int) : The string which will be checked for a question mark

Returns:

True (bool): Whenever the given string is a question

False (bool): Whenever the given string is not a question

Definition at line 745 of file d_create_Big_CSV_ar.py.

def d_create_Big_CSV_ar.check_if_comment_is_not_from_thread_author (*author_of_thread*, *comment_author*)

Checks whether both strings are equal or not

1. This method simply checks whether both strings match each other or not.

I have built this extra method to have a better overview in the main code..

Args:

author_of_thread (str) : The name of the thread author (iAMA-Host)

comment author (str) : The name of the comments author

Returns:

True (bool): Whenever the strings do not match

False (bool): Whenever the strings do match

answered that given question)

Definition at line 782 of file d_create_Big_CSV_ar.py.

def d_create_Big_CSV_ar.check_if_comment_is_on_tier_1 (*comment_parent_id*)

Checks whether a comment relies on the first tier or any other tier

Args:

comment_parent_id (str) : The name id of the comments parent

Returns:

```
True (bool): Whenever the comment lies on tier 1
False (bool): Whenever the comment lies on any other tier
```

Definition at line 766 of file d_create_Big_CSV_ar.py.

def d_create_Big_CSV_ar.check_script_arguments ()

```
Checks if enough and correct arguments have been given to run this script adequate

1. It checks in the first instance if enough arguments have been given
2. Then necessary variables will be filled with appropriate values

Args:
-
Returns:
-
```

Definition at line 20 of file d_create_Big_CSV_ar.py.

def d_create_Big_CSV_ar.generate_data ()

```
Starts calculating various information about thread and iama behaviour related to the year which
is currently
being processed

After the caluclations have every iteration the results will ber appended to a list, which will
contain all that
information for the current year... That list will be writtend to csv and appended to a global
list in other
methods

Args:
-
Returns:
-
```

Definition at line 105 of file d_create_Big_CSV_ar.py.

def d_create_Big_CSV_ar.initialize_mongo_db_parameters (*actually_processed_year*)

```
Instantiates all necessary variables for the correct usage of the mongoDB-Client

Args:
    actually_processed_year (int) : The year with which parameters the database should be accessed
Returns:
-
```

Definition at line 45 of file d_create_Big_CSV_ar.py.

def d_create_Big_CSV_ar.process_specific_thread (*thread_id*, *thread_creation_time_stamp*, *thread_author*)

```
Does the needed operations, for gaining information / knowledge about threads on the given thread
id

After the caluclations have every iteration the results will ber appended to a list, which will
contain all that
information for the current year... That list will be writtend to csv and appended to a global
list in other
```



```

        methods

Args:
    thread_id (str) : The id, needed for operating (i.E. comparison of parent - child relation)
    thread creation time stamp (int) : Creation time stamp of thread, needed for time difference
    calculation
    thread_author (str): The name of the threads author, needed for answer checking of a post
Returns:
    -

```

Definition at line 260 of file d_create_Big_CSV_ar.py.

def d_create_Big_CSV_ar.start_data_generation_for_analysis ()

```

Starts the whole combination of generating data, checking data and writing them into csv files

1. Triggers the data generation process and moves forward within the years -
    by moving through the years a csv file will be created for every year

Args:
    -
Returns:
    -

```

Definition at line 65 of file d_create_Big_CSV_ar.py.

def d_create_Big_CSV_ar.write_csv_data (*list_with_information*)

```

Creates a csv file containing all necessary information about the thread and its mannerism to do
research on

Args:
    list with information (list) : Contains various information about threads mannerism
Returns:
    -

```

Definition at line 1044 of file d_create_Big_CSV_ar.py.

Variable Documentation

int d_create_Big_CSV_ar.argument_year_beginning = 0

Definition at line 1203 of file d_create_Big_CSV_ar.py.

int d_create_Big_CSV_ar.argument_year_ending = 0

Definition at line 1206 of file d_create_Big_CSV_ar.py.

list d_create_Big_CSV_ar.list_current_year = []

Definition at line 1212 of file d_create_Big_CSV_ar.py.

```
list d_create_Big_CSV_ar.list_global_year = []
```

Definition at line 1215 of file d_create_Big_CSV_ar.py.

```
int d_create_Big_CSV_ar.year_actually_in_progress = 0
```

Definition at line 1209 of file d_create_Big_CSV_ar.py.

d_create_Big_CSV_med Namespace Reference

Functions

- def [check_script_arguments](#) ()
- def [initialize_mongo_db_parameters](#) (actually_processed_year)
- def [start_data_generation_for_analysis](#) ()
- def [generate_data](#) ()
- def [process_specific_thread](#) (thread_id, thread_creation_time_stamp, thread_author)
- def [check_if_comment_is_a_question](#) (given_string)
- def [check_if_comment_is_on_tier_1](#) (comment_parent_id)
- def [check_if_comment_is_not_from_thread_author](#) (author_of_thread, comment_author)
- def [check_if_comment_has_been_answered_by_thread_author](#) (author_of_thread, comment_actual_id, comments_cursor)
- def [calculate_time_difference](#) (comment_time_stamp, answer_time_stamp_iama_host)
- def [calculate_reaction_time_average](#) (list_to_be_processed, thread_creation_time_stamp)
- def [calculate_life_span](#) (thread_creation_time_stamp, time_value_of_last_comment, time_value_of_last_question)
- def [add_actual_year_list_to_global_list](#) (list_to_append)
- def [write_csv_data](#) (list_with_information)

Variables

- int [argument_year_beginning](#) = 0
- int [argument_year_ending](#) = 0
- int [year_actually_in_progress](#) = 0
- list [list_current_year](#) = []
- list [list_global_year](#) = []

Function Documentation

def d_create_Big_CSV_med.add_actual_year_list_to_global_list (*list_to_append*)

```
Iterates over a given list with thread information and adds every single element to a global list
The global list will be printed to csv in the end
```

Args:

```
list_to_append (list) : List with thread information which will be appended to a global list
```

Returns:

```
-
```

Definition at line 1028 of file d_create_Big_CSV_med.py.

**def d_create_Big_CSV_med.calculate_life_span (*thread_creation_time_stamp*,
time_value_of_last_comment, *time_value_of_last_question*)**

```
Calculates the life span between to time stamps
```

```
1. The creation date of a thread gets determined
2. Then the comments will be iterated over, creating a dictionary which is structured as follows:
{
    ('first_Comment_After_Thread_Started', int),
    ('thread_life_span', int),
    ('arithmetic_Mean_Response_Time', int),
```

```

        ('median_Response_Time', int),
        ('id')
    }
3. That returned dictionary will be appended to a global list
4. That List will be iterated later on and the appropriate graph will be plotted

Args:
    thread_creation_time_stamp (float) : The time stamp (utc epoch) of the thread creation
    time value of last comment (float) : The time stamp (utc epoch) of the threads last comment
    time_value_of_last_question (float) : The time stamp (utc epoch) of the threads last question
Returns:
    dict_to_be_returned (dict) : Containing information about the time differences:
        Thread creation timestamp <-> Last question time stamp

```

Thread creation timestamp <-> Last comment time stamp

Definition at line 973 of file d_create_Big_CSV_med.py.

**def d_create_Big_CSV_med.calculate_reaction_time_average (*list_to_be_processed*,
thread_creation_time_stamp)**

```

Calculates the reaction time of a list with time values in it

Args:
    list to be processed (list) : The list which contains time values (utc epoch)
    thread_creation_time_stamp (str) : The string which contains the creation date of the thread
    (utc epoch)
Returns:
    None : Whenever there were no time values given

```

np.median(time_difference) (float) : Time arithmetic median of the reaction time in seconds

Definition at line 889 of file d_create_Big_CSV_med.py.

**def d_create_Big_CSV_med.calculate_time_difference (*comment_time_stamp*,
answer_time_stamp_iama_host)**

```

Calculates the time difference in seconds between the a comment and its answer from the iama host

1. The time stamps will be converted from epoch
    into float and afterwards into str again
    (necessary for correct subtraction)
2. Then the time stamps will be subtracted from each other
3. The containing time difference will be converted into seconds (int)

Args:
    comment_time_stamp (str): The time stamp of the comment
    answer_time_stamp_iama_host (str): The time stamp of the iAMA hosts answer
Returns:
    time_difference_in_seconds (int) : The time difference of the comment and its answer by the
    iAMA host in seconds

```

Definition at line 850 of file d_create_Big_CSV_med.py.

def d_create_Big_CSV_med.check_if_comment_has_been_answered_by_thread_author (
***author_of_thread*, *comment_actual_id*, *comments_cursor*)**

```

Checks whether both strings are equal or not

1. A dictionary containing flags whether that a question is answered by the host with the appropriate
    timestamp will
    be created in the beginning.
2. Then the method iterates over every comment within that thread
    1.1. Whenever an answer is from the iAMA hosts and the processed comments 'parent_id' matches
    the iAMA hosts

```

```

        comments (answers) id, the returned dict will contain appropriate values and will be
        returned
    1.2. If this is not the case, it will be returned in its default condition

Note: We take a list as 'comments cursor' and not a real cursor, because real cursors can be
exhausted, which
    could lead to, that not all comments will be iterated.. This is especially critical when
you have to do
    many iterations with only one cursor... [took me 8 hours to figure this "bug" out...]

Args:
    author_of_thread (str) : The name of the thread author (iAMA-Host)
    comment_actual_id (str) : The id of the actually processed comment
    comments_cursor (list) : The list containing all comments
Returns:
    True (bool): Whenever the strings do not match
    False (bool): Whenever the strings do match
    answered that given question)

```

Definition at line 802 of file d_create_Big_CSV_med.py.

def d_create_Big_CSV_med.check_if_comment_is_a_question (*given_string*)

```

Simply checks whether a given string is a question or not

This method simply checks whether a question mark exists within that string or not..
    This is just that simple because messing around with natural processing kits to determine the
semantic sense
    would blow up my bachelor work...

Args:
    given_string (int) : The string which will be checked for a question mark
Returns:
    True (bool): Whenever the given string is a question
    False (bool): Whenever the given string is not a question

```

Definition at line 745 of file d_create_Big_CSV_med.py.

**def d_create_Big_CSV_med.check_if_comment_is_not_from_thread_author (*author_of_thread*,
comment_author)**

```

Checks whether both strings are equal or not

1. This method simply checks whether both strings match each other or not.
    I have built this extra method to have a better overview in the main code..

Args:
    author_of_thread (str) : The name of the thread author (iAMA-Host)
    comment author (str) : The name of the comments author
Returns:
    True (bool): Whenever the strings do not match
    False (bool): Whenever the strings do match
    answered that given question)

```

Definition at line 782 of file d_create_Big_CSV_med.py.

def d_create_Big_CSV_med.check_if_comment_is_on_tier_1 (*comment_parent_id*)

```

Checks whether a comment relies on the first tier or any other tier

Args:
    comment_parent_id (str) : The name id of the comments parent
Returns:

```

```
True (bool): Whenever the comment lies on tier 1
False (bool): Whenever the comment lies on any other tier
```

Definition at line 766 of file d_create_Big_CSV_med.py.

def d_create_Big_CSV_med.check_script_arguments ()

```
Checks if enough and correct arguments have been given to run this script adequate

1. It checks in the first instance if enough arguments have been given
2. Then necessary variables will be filled with appropriate values

Args:
-
Returns:
-
```

Definition at line 20 of file d_create_Big_CSV_med.py.

def d_create_Big_CSV_med.generate_data ()

```
Starts calculating various information about thread and iama behaviour related to the year which
is currently
being processed

After the caluclations have every iteration the results will ber appended to a list, which will
contain all that
information for the current year... That list will be writtend to csv and appended to a global
list in other
methods

Args:
-
Returns:
-
```

Definition at line 105 of file d_create_Big_CSV_med.py.

def d_create_Big_CSV_med.initialize_mongo_db_parameters (*actually_processed_year*)

```
Instantiates all necessary variables for the correct usage of the mongoDB-Client

Args:
    actually_processed_year (int) : The year with which parameters the database should be accessed
Returns:
-
```

Definition at line 45 of file d_create_Big_CSV_med.py.

def d_create_Big_CSV_med.process_specific_thread (*thread_id*, *thread_creation_time_stamp*, *thread_author*)

```
Does the needed operations, for gaining information / knowledge about threads on the given thread
id

After the caluclations have every iteration the results will ber appended to a list, which will
contain all that
information for the current year... That list will be writtend to csv and appended to a global
list in other
```

```

        methods

Args:
    thread_id (str) : The id, needed for operating (i.E. comparison of parent - child relation)
    thread creation time stamp (int) : Creation time stamp of thread, needed for time difference
    calculation
    thread_author (str): The name of the threads author, needed for answer checking of a post
Returns:
    -

```

Definition at line 260 of file d_create_Big_CSV_med.py.

def d_create_Big_CSV_med.start_data_generation_for_analysis ()

```

Starts the whole combination of generating data, checking data and writing them into csv files

1. Triggers the data generation process and moves forward within the years -
    by moving through the years a csv file will be created for every year

Args:
    -
Returns:
    -

```

Definition at line 65 of file d_create_Big_CSV_med.py.

def d_create_Big_CSV_med.write_csv_data (*list_with_information*)

```

Creates a csv file containing all necessary information about the thread and its mannerism to do
research on

Args:
    list with information (list) : Contains various information about threads mannerism
Returns:
    -

```

Definition at line 1044 of file d_create_Big_CSV_med.py.

Variable Documentation

int d_create_Big_CSV_med.argument_year_beginning = 0

Definition at line 1203 of file d_create_Big_CSV_med.py.

int d_create_Big_CSV_med.argument_year_ending = 0

Definition at line 1206 of file d_create_Big_CSV_med.py.

list d_create_Big_CSV_med.list_current_year = []

Definition at line 1212 of file d_create_Big_CSV_med.py.

```
list d_create_Big_CSV_med.list_global_year = []
```

Definition at line 1215 of file d_create_Big_CSV_med.py.

```
int d_create_Big_CSV_med.year_actually_in_progress = 0
```

Definition at line 1209 of file d_create_Big_CSV_med.py.

PlotlyBarChart Namespace Reference

Classes

- class [PlotlyBarChart](#)

PlotlyBarChart_5_Bars Namespace Reference

Classes

- class [PlotlyBarChart5Bars](#)

Class Documentation

PlotlyBarChart.PlotlyBarChart Class Reference

Public Member Functions

- def [__init__](#) (self)
- def [main_method](#) (self, list_of_calculated_data)

Static Public Member Functions

- def [fill_x_axis_list](#) (list_of_calculated_data)
- def [fill_y_axis_answered_list](#) (list_of_calculated_data)
- def [fill_y_axis_unanswered_list](#) (list_of_calculated_data)
- def [fill_bar_percentages_values](#) (list_of_calculated_data)
- def [fill_chart_title_description](#) (list_of_calculated_data)
- def [fill_bar_description](#) (list_of_calculated_data)
- def [generate_chart](#) ()

Static Public Attributes

- [time_now_date](#) = time.strftime("%d.%m.%Y")
- [time_now_time](#) = time.strftime("%H:%M:%S")
- string [bar_x_axis_text](#) = 'Chart creation date: '
- string [chart_title](#) = ""
- list [bar_value_description](#) = []
- list [bar_x_axis_values](#) = []
- list [bar_y_axis_first_values](#) = []
- list [bar_y_axis_second_values](#) = []
- list [bar_first_n_second_values_percentage](#) = []

Detailed Description

```
The class to create a stacked bar chart.  
This class is heavily modified because it pyplot normally is not designed to run offline this way..
```

```
Args:  
-  
Returns:  
-
```

Definition at line 13 of file PlotlyBarChart.py.

Constructor & Destructor Documentation

def PlotlyBarChart.PlotlyBarChart.__init__ (self)

```
Instantiates the class  
Args:  
-
```

```
Returns:  
-
```

Definition at line 44 of file PlotlyBarChart.py.

Member Function Documentation

def PlotlyBarChart.PlotlyBarChart.fill_bar_description (*list_of_calculated_data*)[static]

```
Defines the bar description in dependence to given parameters list_of_calculated_data[0][0]  
  
Args:  
    list_of_calculated_data (list) : Will be accessed to gain necessary values  
Returns:  
-
```

Definition at line 208 of file PlotlyBarChart.py.

def PlotlyBarChart.PlotlyBarChart.fill_bar_percentages_values (*list_of_calculated_data*)[static]

```
Calculates percentages to be shown within the graph..  
    This is not supported within pyplot under normal circumstances.. so we're tricking the HTML  
    settings  
  
Args:  
    list_of_calculated_data (list) : Will be iterated to gain necessary values  
Returns:  
-
```

Definition at line 129 of file PlotlyBarChart.py.

def PlotlyBarChart.PlotlyBarChart.fill_chart_title_description (*list_of_calculated_data*)[static]

```
Defines the chart title in dependence to sorting method and processed years  
  
Args:  
    list_of_calculated_data (list) : Will be accessed to gain necessary values  
Returns:  
-
```

Definition at line 160 of file PlotlyBarChart.py.

def PlotlyBarChart.PlotlyBarChart.fill_x_axis_list (*list_of_calculated_data*)[static]

```
Fills the "x axis" with the values of the years  
  
Args:  
    list of calculated data (list) : Will be iterated to gain necessary values  
Returns:  
-
```

Definition at line 81 of file PlotlyBarChart.py.

```
def PlotlyBarChart.PlotlyBarChart.fill_y_axis_answered_list ( list_of_calculated_data)[static]
```

```
Fills an bar within the chart with values of the amount of unanswered questions  
  
Args:  
    list_of_calculated_data (list) : Will be iterated to gain necessary values  
Returns:  
    -
```

Definition at line 97 of file PlotlyBarChart.py.

```
def PlotlyBarChart.PlotlyBarChart.fill_y_axis_unanswered_list (  
list_of_calculated_data)[static]
```

```
Fills an bar within the chart with values of the amount of unanswered questions  
  
Args:  
    list_of_calculated_data (list) : Will be iterated to gain necessary values  
Returns:  
    -
```

Definition at line 113 of file PlotlyBarChart.py.

```
def PlotlyBarChart.PlotlyBarChart.generate_chart ()[static]
```

```
Generates the chart "temp-plot.html" which will be automatically opened within the browser  
  
Args:  
    -  
Returns:  
    -
```

Definition at line 235 of file PlotlyBarChart.py.

```
def PlotlyBarChart.PlotlyBarChart.main_method ( self, list_of_calculated_data)
```

```
Sequential fills the necessary variables for the graph  
Structure of list of calculated data:  
  
[ "sorting", [year, answered, unanswered], [year, answered, unanswered], ... ]  
i.e. ["top",  
      [2009, 900, 1536],  
      [2010, 500, 500],  
      [2011, 300, 700]  
      ]  
  
Args:  
    list_of_calculated_data (list): Contains sorting method, and the years data  
Returns:  
    -
```

Definition at line 55 of file PlotlyBarChart.py.

Member Data Documentation

list PlotlyBarChart.PlotlyBarChart.bar_first_n_second_values_percentage = [] [static]

Definition at line 42 of file PlotlyBarChart.py.

list PlotlyBarChart.PlotlyBarChart.bar_value_description = [] [static]

Definition at line 32 of file PlotlyBarChart.py.

string PlotlyBarChart.PlotlyBarChart.bar_x_axis_text = 'Chart creation date: ' [static]

Definition at line 26 of file PlotlyBarChart.py.

list PlotlyBarChart.PlotlyBarChart.bar_x_axis_values = [] [static]

Definition at line 33 of file PlotlyBarChart.py.

list PlotlyBarChart.PlotlyBarChart.bar_y_axis_first_values = [] [static]

Definition at line 36 of file PlotlyBarChart.py.

list PlotlyBarChart.PlotlyBarChart.bar_y_axis_second_values = [] [static]

Definition at line 39 of file PlotlyBarChart.py.

string PlotlyBarChart.PlotlyBarChart.chart_title = "" [static]

Definition at line 29 of file PlotlyBarChart.py.

PlotlyBarChart.PlotlyBarChart.time_now_date = time.strftime("%d.%m.%Y") [static]

Definition at line 23 of file PlotlyBarChart.py.

PlotlyBarChart.PlotlyBarChart.time_now_time = time.strftime("%H:%M:%S") [static]

Definition at line 24 of file PlotlyBarChart.py.

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

- [PlotlyBarChart.py](#)

PlotlyBarChart_5_Bars.PlotlyBarChart5Bars Class Reference

Public Member Functions

- def [__init__](#) (self)
- def [main_method](#) (self, list_of_calculated_data)

Static Public Member Functions

- def [fill_x_axis_list](#) (list_of_calculated_data)
- def [fill_y_axis_values](#) (list_of_calculated_data)
- def [fill_bar_percentages_values](#) (list_of_calculated_data)
- def [fill_chart_title_description](#) (list_of_calculated_data)
- def [fill_bar_description](#) (list_of_calculated_data)
- def [fill_bar_annotations](#) ()
- def [generate_chart](#) ()

Static Public Attributes

- string [color_1](#) = 'rgba(255, 114, 86, 1.0)'
 - string [color_1_border](#) = 'rgba(238, 106, 80, 1.0)'
 - string [color_2](#) = 'rgba(238, 118, 0, 1.0)'
 - string [color_2_border](#) = 'rgba(205, 102, 0, 1.0)'
 - string [color_3](#) = 'rgba(0, 201, 87, 1.0)'
 - string [color_3_border](#) = 'rgba(0, 139, 0, 1.0)'
 - string [color_4](#) = 'rgba(0, 205, 205, 1.0)'
 - string [color_4_border](#) = 'rgba(0, 139, 139, 1.0)'
 - string [color_5](#) = 'rgba(137, 104, 205, 1.0)'
 - string [color_5_border](#) = 'rgba(39, 71, 139, 1.0)'
 - [time_now_date](#) = time.strftime("%d.%m.%Y")
 - [time_now_time](#) = time.strftime("%H:%M:%S")
 - string [bar_x_axis_text](#) = 'Chart creation date: '
 - string [chart_title](#) = ""
 - list [bar_value_description](#) = []
 - list [bar_x_axis_values](#) = []
 - list [bar_y_axis_first_values](#) = []
 - list [bar_y_axis_second_values](#) = []
 - list [bar_y_axis_third_values](#) = []
 - list [bar_y_axis_fourth_values](#) = []
 - list [bar_y_axis_fifth_values](#) = []
 - list [bar_percentages_values_1](#) = []
 - list [bar_percentages_values_2](#) = []
 - list [bar_percentages_values_3](#) = []
 - list [bar_percentages_values_4](#) = []
 - list [bar_percentages_values_5](#) = []
 - list [annotations_1](#) = []
 - list [annotations_2](#) = []
 - list [annotations_3](#) = []
 - list [annotations_4](#) = []
 - list [annotations_5](#) = []
 - list [annotations_all](#) = []
-

Detailed Description

```
The class to create a stacked bar chart.  
    This class is heavily modified because it pyplot normally is not designed to run offline this way..  
  
Args:  
    -  
Returns:  
    -
```

Definition at line 13 of file PlotlyBarChart_5_Bars.py.

Constructor & Destructor Documentation

def PlotlyBarChart_5_Bars.PlotlyBarChart5Bars.__init__ (self)

```
Instanciates the class  
  
Args:  
    -  
Returns:  
    -
```

Definition at line 71 of file PlotlyBarChart_5_Bars.py.

Member Function Documentation

def PlotlyBarChart_5_Bars.PlotlyBarChart5Bars.fill_bar_annotations () [static]

Definition at line 291 of file PlotlyBarChart_5_Bars.py.

def PlotlyBarChart_5_Bars.PlotlyBarChart5Bars.fill_bar_description (list_of_calculated_data) [static]

```
Defines the bar description in dependence to given parameters list_of_calculated_data[0][0]  
  
Args:  
    list_of_calculated_data (list) : Will be accessed to gain necessary values  
Returns:  
    -
```

Definition at line 246 of file PlotlyBarChart_5_Bars.py.

def PlotlyBarChart_5_Bars.PlotlyBarChart5Bars.fill_bar_percentages_values (list_of_calculated_data) [static]

```
Calculates percentages to be shown within the graph..  
    This is not supported within pyplot under normal circumstances.. so we're tricking the HTML  
settings  
  
Args:  
    list_of_calculated_data (list) : Will be iterated to gain necessary values
```



```
Returns:  
-
```

Definition at line 144 of file PlotlyBarChart_5_Bars.py.

```
def PlotlyBarChart_5_Bars.PlotlyBarChart5Bars.fill_chart_title_description (  
list_of_calculated_data)[static]
```

```
Defines the chart title in dependence to sorting method and processed years  
  
Args:  
    list_of_calculated_data (list) : Will be accessed to gain necessary values  
Returns:  
-
```

Definition at line 189 of file PlotlyBarChart_5_Bars.py.

```
def PlotlyBarChart_5_Bars.PlotlyBarChart5Bars.fill_x_axis_list (  
list_of_calculated_data)[static]
```

```
Fills the "x axis" with the values of the years  
  
Args:  
    list of calculated data (list) : Will be iterated to gain necessary values  
Returns:  
-
```

Definition at line 108 of file PlotlyBarChart_5_Bars.py.

```
def PlotlyBarChart_5_Bars.PlotlyBarChart5Bars.fill_y_axis_values (  
list_of_calculated_data)[static]
```

```
Fills an bar within the chart with values of the amount of unanswered questions  
  
Args:  
    list of calculated data (list) : Will be iterated to gain necessary values  
Returns:  
-
```

Definition at line 124 of file PlotlyBarChart_5_Bars.py.

```
def PlotlyBarChart_5_Bars.PlotlyBarChart5Bars.generate_chart ()[static]
```

```
Generates the chart "temp-plot.html" which will be automatically opened within the browser  
  
Args:  
-  
Returns:  
-
```

Definition at line 393 of file PlotlyBarChart_5_Bars.py.

```
def PlotlyBarChart_5_Bars.PlotlyBarChart5Bars.main_method ( self, list_of_calculated_data)
```

```

Sequential fills the necessary variables for the graph
Structure of list_of_calculated_data:

[ "sorting", [year, answered, unanswered], [year, answered, unanswered], ... ]
i.e. ["top",
      [2009, 900, 1536],
      [2010, 500, 500],
      [2011, 300, 700]
      ]

Args:
    list_of_calculated_data (list): Contains sorting method, and the years data
Returns:
    -

```

Definition at line 82 of file PlotlyBarChart_5_Bars.py.

Member Data Documentation

list PlotlyBarChart_5_Bars.PlotlyBarChart5Bars.annotations_1 = [] [static]

Definition at line 64 of file PlotlyBarChart_5_Bars.py.

list PlotlyBarChart_5_Bars.PlotlyBarChart5Bars.annotations_2 = [] [static]

Definition at line 65 of file PlotlyBarChart_5_Bars.py.

list PlotlyBarChart_5_Bars.PlotlyBarChart5Bars.annotations_3 = [] [static]

Definition at line 66 of file PlotlyBarChart_5_Bars.py.

list PlotlyBarChart_5_Bars.PlotlyBarChart5Bars.annotations_4 = [] [static]

Definition at line 67 of file PlotlyBarChart_5_Bars.py.

list PlotlyBarChart_5_Bars.PlotlyBarChart5Bars.annotations_5 = [] [static]

Definition at line 68 of file PlotlyBarChart_5_Bars.py.

list PlotlyBarChart_5_Bars.PlotlyBarChart5Bars.annotations_all = [] [static]

Definition at line 69 of file PlotlyBarChart_5_Bars.py.

list PlotlyBarChart_5_Bars.PlotlyBarChart5Bars.bar_percentages_values_1 = [] [static]

Definition at line 58 of file PlotlyBarChart_5_Bars.py.

list PlotlyBarChart_5_Bars.PlotlyBarChart5Bars.bar_percentages_values_2 = [] [static]

Definition at line 59 of file PlotlyBarChart_5_Bars.py.

```
list PlotlyBarChart_5_Bars.PlotlyBarChart5Bars.bar_percentages_values_3 = [] [static]
```

Definition at line 60 of file PlotlyBarChart_5_Bars.py.

```
list PlotlyBarChart_5_Bars.PlotlyBarChart5Bars.bar_percentages_values_4 = [] [static]
```

Definition at line 61 of file PlotlyBarChart_5_Bars.py.

```
list PlotlyBarChart_5_Bars.PlotlyBarChart5Bars.bar_percentages_values_5 = [] [static]
```

Definition at line 62 of file PlotlyBarChart_5_Bars.py.

```
list PlotlyBarChart_5_Bars.PlotlyBarChart5Bars.bar_value_description = [] [static]
```

Definition at line 47 of file PlotlyBarChart_5_Bars.py.

```
string PlotlyBarChart_5_Bars.PlotlyBarChart5Bars.bar_x_axis_text = 'Chart creation date:  
' [static]
```

Definition at line 41 of file PlotlyBarChart_5_Bars.py.

```
list PlotlyBarChart_5_Bars.PlotlyBarChart5Bars.bar_x_axis_values = [] [static]
```

Definition at line 49 of file PlotlyBarChart_5_Bars.py.

```
list PlotlyBarChart_5_Bars.PlotlyBarChart5Bars.bar_y_axis_fifth_values = [] [static]
```

Definition at line 55 of file PlotlyBarChart_5_Bars.py.

```
list PlotlyBarChart_5_Bars.PlotlyBarChart5Bars.bar_y_axis_first_values = [] [static]
```

Definition at line 51 of file PlotlyBarChart_5_Bars.py.

```
list PlotlyBarChart_5_Bars.PlotlyBarChart5Bars.bar_y_axis_fourth_values = [] [static]
```

Definition at line 54 of file PlotlyBarChart_5_Bars.py.

```
list PlotlyBarChart_5_Bars.PlotlyBarChart5Bars.bar_y_axis_second_values = [] [static]
```

Definition at line 52 of file PlotlyBarChart_5_Bars.py.

```
list PlotlyBarChart_5_Bars.PlotlyBarChart5Bars.bar_y_axis_third_values = [] [static]
```

Definition at line 53 of file PlotlyBarChart_5_Bars.py.

```
string PlotlyBarChart_5_Bars.PlotlyBarChart5Bars.chart_title = "" [static]
```

Definition at line 44 of file PlotlyBarChart_5_Bars.py.

```
string PlotlyBarChart_5_Bars.PlotlyBarChart5Bars.color_1 = 'rgba(255, 114, 86, 1.0)' [static]
```

Definition at line 23 of file PlotlyBarChart_5_Bars.py.

```
string PlotlyBarChart_5_Bars.PlotlyBarChart5Bars.color_1_border = 'rgba(238, 106, 80, 1.0)' [static]
```

Definition at line 24 of file PlotlyBarChart_5_Bars.py.

```
string PlotlyBarChart_5_Bars.PlotlyBarChart5Bars.color_2 = 'rgba(238, 118, 0, 1.0)' [static]
```

Definition at line 26 of file PlotlyBarChart_5_Bars.py.

```
string PlotlyBarChart_5_Bars.PlotlyBarChart5Bars.color_2_border = 'rgba(205, 102, 0, 1.0)' [static]
```

Definition at line 27 of file PlotlyBarChart_5_Bars.py.

```
string PlotlyBarChart_5_Bars.PlotlyBarChart5Bars.color_3 = 'rgba(0, 201, 87, 1.0)' [static]
```

Definition at line 29 of file PlotlyBarChart_5_Bars.py.

```
string PlotlyBarChart_5_Bars.PlotlyBarChart5Bars.color_3_border = 'rgba(0, 139, 0, 1.0)' [static]
```

Definition at line 30 of file PlotlyBarChart_5_Bars.py.

```
string PlotlyBarChart_5_Bars.PlotlyBarChart5Bars.color_4 = 'rgba(0, 205, 205, 1.0)' [static]
```

Definition at line 32 of file PlotlyBarChart_5_Bars.py.

```
string PlotlyBarChart_5_Bars.PlotlyBarChart5Bars.color_4_border = 'rgba(0, 139, 139, 1.0)' [static]
```

Definition at line 33 of file PlotlyBarChart_5_Bars.py.

```
string PlotlyBarChart_5_Bars.PlotlyBarChart5Bars.color_5 = 'rgba(137, 104, 205, 1.0)' [static]
```

Definition at line 35 of file PlotlyBarChart_5_Bars.py.

```
string PlotlyBarChart_5_Bars.PlotlyBarChart5Bars.color_5_border = 'rgba(39, 71, 139, 1.0)'[static]
```

Definition at line 36 of file PlotlyBarChart_5_Bars.py.

```
PlotlyBarChart_5_Bars.PlotlyBarChart5Bars.time_now_date = time.strftime("%d.%m.%Y")[static]
```

Definition at line 38 of file PlotlyBarChart_5_Bars.py.

```
PlotlyBarChart_5_Bars.PlotlyBarChart5Bars.time_now_time = time.strftime("%H:%M:%S")[static]
```

Definition at line 39 of file PlotlyBarChart_5_Bars.py.

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

- [PlotlyBarChart_5_Bars.py](#)

File Documentation

a__everything_Big_CSV_analyzer.py File Reference

Namespaces

- [a__everything_Big_CSV_analyzer](#)

Functions

- def [a__everything_Big_CSV_analyzer.relation_question_upvotes_with_amount_of_questions_answered_by_iamahost\(\)](#)
- def [a__everything_Big_CSV_analyzer.average_means_of_values_f_threads\(\)](#)
- def [a__everything_Big_CSV_analyzer.relation_thread_upvotes_with_amount_of_comments\(\)](#)
- def [a__everything_Big_CSV_analyzer.relation_thread_upvotes_with_amount_of_questions\(\)](#)
- def [a__everything_Big_CSV_analyzer.relation_thread_downvotes_with_amount_of_comments\(\)](#)
- def [a__everything_Big_CSV_analyzer.relation_thread_downvotes_with_amount_of_questions\(\)](#)
- def [a__everything_Big_CSV_analyzer.relation_thread_upvotes_and_iamahost_response_time_comments\(\)](#)
- def [a__everything_Big_CSV_analyzer.relation_thread_upvotes_and_iamahost_response_time_questions\(\)](#)
- def [a__everything_Big_CSV_analyzer.relation_thread_downvotes_and_iamahost_response_time_comments\(\)](#)
- def [a__everything_Big_CSV_analyzer.relation_thread_downvotes_and_iamahost_response_time_questions\(\)](#)
- def [a__everything_Big_CSV_analyzer.relation_thread_lifespan_to_last_comment_and_amount_of_comments\(\)](#)
- def [a__everything_Big_CSV_analyzer.relation_thread_lifespan_to_last_comment_and_amount_of_questions\(\)](#)
- def [a__everything_Big_CSV_analyzer.relation_thread_lifespan_to_last_question_and_amount_of_comments\(\)](#)
- def [a__everything_Big_CSV_analyzer.relation_thread_lifespan_to_last_question_and_amount_of_question\(\)](#)
- def [a__everything_Big_CSV_analyzer.relation_thread_lifespan_to_last_comment_and_iamahost_response_time_to_comments\(\)](#)
- def [a__everything_Big_CSV_analyzer.relation_thread_lifespan_to_last_comment_and_iamahost_response_time_to_questions\(\)](#)
- def [a__everything_Big_CSV_analyzer.relation_thread_lifespan_to_last_question_and_iamahost_response_time_to_comments\(\)](#)
- def [a__everything_Big_CSV_analyzer.relation_thread_lifespan_to_last_question_and_iamahost_response_time_to_questions\(\)](#)
- def [a__everything_Big_CSV_analyzer.relation_thread_reaction_time_comments_and_iamahost_response_time_to_comments\(\)](#)
- def [a__everything_Big_CSV_analyzer.relation_thread_reaction_time_comments_and_iamahost_response_time_to_questions\(\)](#)
- def [a__everything_Big_CSV_analyzer.relation_thread_reaction_time_questions_and_iamahost_response_time_to_comments\(\)](#)
- def [a__everything_Big_CSV_analyzer.relation_thread_reaction_time_questions_and_iamahost_response_time_to_questions\(\)](#)
- def [a__everything_Big_CSV_analyzer.relation_thread_reaction_time_comments_and_amount_of_comments_the_iamahost_answered_to\(\)](#)

- [def a_everything_Big_CSV_analyzer.relation_thread_reaction_time_comments_and_amount_of_questions_the_ia_ma_host_answered_to\(\)](#)
- [def a_everything_Big_CSV_analyzer.relation_thread_reaction_time_questions_and_amount_of_comments_the_ia_ma_host_answered_to\(\)](#)
- [def a_everything_Big_CSV_analyzer.relation_thread_reaction_time_questions_and_amount_of_questions_the_ia_ma_host_answered_to\(\)](#)
- [def a_everything_Big_CSV_analyzer.relation_thread_amount_of_questioners_total_and_num_questions_answered_by_ia_ma_host\(\)](#)
- [def a_everything_Big_CSV_analyzer.relation_thread_amount_of_commentators_total_and_num_comments_answered_by_ia_ma_host\(\)](#)
- [def a_everything_Big_CSV_analyzer.relation_thread_amount_of_questions_and_amount_questions_answered_by_ia_ma_host\(\)](#)
- [def a_everything_Big_CSV_analyzer.thread_overall_correlation\(\)](#)
- [def a_everything_Big_CSV_analyzer.question_overall_correlation\(\)](#)
- [def a_everything_Big_CSV_analyzer.average_means_of_values_f_authors\(\)](#)
- [def a_everything_Big_CSV_analyzer.median_of_values_f_authors_randomized\(\)](#)
- [def a_everything_Big_CSV_analyzer.median_of_values_f_authors\(\)](#)
- [def a_everything_Big_CSV_analyzer.arr_of_values_f_authors\(\)](#)
- [def a_everything_Big_CSV_analyzer.calculate_t_tests_of_author_values\(\)](#)

Variables

- [a_everything_Big_CSV_analyzer.author_information_ia_ma](#)
- [a_everything_Big_CSV_analyzer.author_information_ia_ma_sampleset_randomized](#)
- [a_everything_Big_CSV_analyzer.author_information_random](#)
- [a_everything_Big_CSV_analyzer.author_information_random_sampleset_randomized](#)
- [a_everything_Big_CSV_analyzer.thread_information](#)
- [a_everything_Big_CSV_analyzer.question_information](#)
- [a_everything_Big_CSV_analyzer.inplace](#)
- [a_everything_Big_CSV_analyzer.author_amount_creation_ia_ma_threads = author_information_ia_ma\['amount_creation_ia_ma_threads'\]](#)
- [a_everything_Big_CSV_analyzer.author_amount_creation_other_threads = author_information_ia_ma\['amount_creation_other_threads'\]](#)
- [a_everything_Big_CSV_analyzer.author_amount_of_comments_except_ia_ma = author_information_ia_ma\['amount_of_comments_except_ia_ma'\]](#)
- [a_everything_Big_CSV_analyzer.author_amount_of_comments_ia_ma = author_information_ia_ma\['amount_of_comments_ia_ma'\]](#)
- [a_everything_Big_CSV_analyzer.author_author_birth_date = author_information_ia_ma\['author_birth_date'\]](#)
- [a_everything_Big_CSV_analyzer.author_author_comment_karma_amount = author_information_ia_ma\['author_comment_karma_amount'\]](#)
- [a_everything_Big_CSV_analyzer.author_author_link_karma_amount = author_information_ia_ma\['author_link_karma_amount'\]](#)
- [a_everything_Big_CSV_analyzer.author_author_name = author_information_ia_ma\['author_name'\]](#)
- [a_everything_Big_CSV_analyzer.author_comment_creation_every_x_sec = author_information_ia_ma\['comment_creation_every_x_sec'\]](#)
- [a_everything_Big_CSV_analyzer.author_thread_creation_every_x_sec = author_information_ia_ma\['thread_creation_every_x_sec'\]](#)
- [a_everything_Big_CSV_analyzer.author_time_acc_birth_first_ia_ma_thread = author_information_ia_ma\['time_acc_birth_first_ia_ma_thread'\]](#)

- [a everything Big CSV analyzer.author time diff acc creation n first comment](#) = author_information_iamas['time_diff_acc_creation_n_first_comment']
- [a everything Big CSV analyzer.author time diff acc creation n first thread](#) = author_information_iamas['time_diff_acc_creation_n_first_thread']
- [a everything Big CSV analyzer.author amount creation iama threads randomized](#) = author_information_iamasampleset_randomized['amount_creation_iamas_threads']
- [a everything Big CSV analyzer.author amount creation other threads randomized](#) = author_information_iamasampleset_randomized['amount_creation_other_threads']
- [a everything Big CSV analyzer.author amount of comments except iama randomized](#) = author_information_iamasampleset_randomized['amount_of_comments_except_iamas']
- [a everything Big CSV analyzer.author amount of comments iama randomized](#) = author_information_iamasampleset_randomized['amount_of_comments_iamas']
- [a everything Big CSV analyzer.author author birth date randomized](#) = author_information_iamasampleset_randomized['author_birth_date']
- [a everything Big CSV analyzer.author author comment karma amount randomized](#) = author_information_iamasampleset_randomized['author_comment_karma_amount']
- [a everything Big CSV analyzer.author author link karma amount randomized](#) = author_information_iamasampleset_randomized['author_link_karma_amount']
- [a everything Big CSV analyzer.author author name randomized](#) = author_information_iamasampleset_randomized['author_name']
- [a everything Big CSV analyzer.author comment creation every x sec randomized](#) = author_information_iamasampleset_randomized['comment_creation_every_x_sec']
- [a everything Big CSV analyzer.author thread creation every x sec randomized](#) = author_information_iamasampleset_randomized['thread_creation_every_x_sec']
- [a everything Big CSV analyzer.author time acc birth first iama thread randomized](#) = author_information_iamasampleset_randomized['time_acc_birth_first_iamas_thread']
- [a everything Big CSV analyzer.author time diff acc creation n first comment randomized](#) = author_information_iamasampleset_randomized['time_diff_acc_creation_n_first_comment']
- [a everything Big CSV analyzer.author time diff acc creation n first thread randomized](#) = author_information_iamasampleset_randomized['time_diff_acc_creation_n_first_thread']
- [a everything Big CSV analyzer.random author amount creation iama threads](#) = author_information_random['amount_creation_iamas_threads']
- [a everything Big CSV analyzer.random author amount creation other threads](#) = author_information_random['amount_creation_other_threads']
- [a everything Big CSV analyzer.random author amount of comments except iama](#) = author_information_random['amount_of_comments_except_iamas']
- [a everything Big CSV analyzer.random author amount of comments iama](#) = author_information_random['amount_of_comments_iamas']
- [a everything Big CSV analyzer.random author author birth date](#) = author_information_random['author_birth_date']
- [a everything Big CSV analyzer.random author author comment karma amount](#) = author_information_random['author_comment_karma_amount']
- [a everything Big CSV analyzer.random author author link karma amount](#) = author_information_random['author_link_karma_amount']
- [a everything Big CSV analyzer.random author author name](#) = author_information_random['author_name']
- [a everything Big CSV analyzer.random author comment creation every x sec](#) = author_information_random['comment_creation_every_x_sec']
- [a everything Big CSV analyzer.random author thread creation every x sec](#) = author_information_random['thread_creation_every_x_sec']
- [a everything Big CSV analyzer.random author time acc birth first iama thread](#) = author_information_random['time_acc_birth_first_iamas_thread']
- [a everything Big CSV analyzer.random author time diff acc creation n first comment](#) = \
- [a everything Big CSV analyzer.random author time diff acc creation n first thread](#)
- [a everything Big CSV analyzer.random author amount creation iama threads randomized](#) = author_information_randomsampleset_randomized['amount_creation_iamas_threads']

- [a everything Big CSV analyzer.random author amount creation other threads randomized](#) = author_information_random_sampleset_randomized['amount_creation_other_threads']
- [a everything Big CSV analyzer.random author amount of comments except iama randomized](#) = author_information_random_sampleset_randomized['amount_of_comments_except_iama']
- [a everything Big CSV analyzer.random author amount of comments iama randomized](#) = author_information_random_sampleset_randomized['amount_of_comments_iama']
- [a everything Big CSV analyzer.random author author birth date randomized](#) = author_information_random_sampleset_randomized['author_birth_date']
- [a everything Big CSV analyzer.random author author comment karma amount randomized](#) = author_information_random_sampleset_randomized['author_comment_karma_amount']
- [a everything Big CSV analyzer.random author author link karma amount randomized](#) = author_information_random_sampleset_randomized['author_link_karma_amount']
- [a everything Big CSV analyzer.random author author name randomized](#) = author_information_random_sampleset_randomized['author_name']
- [a everything Big CSV analyzer.random author comment creation every x sec randomized](#) = author_information_random_sampleset_randomized['comment_creation_every_x_sec']
- [a everything Big CSV analyzer.random author thread creation every x sec randomized](#) = author_information_random_sampleset_randomized['thread_creation_every_x_sec']
- [a everything Big CSV analyzer.random author time acc birth first iama thread randomized](#) = author_information_random_sampleset_randomized['time_acc_birth_first_iama_thread']
- [a everything Big CSV analyzer.random author time diff acc creation n first comment randomized](#) = \
- [a everything Big CSV analyzer.random author time diff acc creation n first thread randomized](#)
- [a everything Big CSV analyzer.question ups](#) = question_information['Question ups']
- [a everything Big CSV analyzer.question answered by iAMA host](#) = question_information['Question answered by iAMA host']
- [a everything Big CSV analyzer.thread year](#) = thread_information['Year']
- [a everything Big CSV analyzer.thread id](#) = thread_information['Thread id']
- [a everything Big CSV analyzer.thread author](#) = thread_information['Thread author']
- [a everything Big CSV analyzer.thread ups](#) = thread_information['Thread ups']
- [a everything Big CSV analyzer.thread downs](#) = thread_information['Thread downs']
- [a everything Big CSV analyzer.thread creation time stamp](#) = thread_information['Thread creation time stamp']
- [a everything Big CSV analyzer.thread average comment vote score total](#)
- [a everything Big CSV analyzer.thread average comment vote score tier 1](#)
- [a everything Big CSV analyzer.thread average comment vote score tier x](#)
- [a everything Big CSV analyzer.thread average question vote score total](#)
- [a everything Big CSV analyzer.thread average question vote score tier 1](#)
- [a everything Big CSV analyzer.thread average question vote score tier x](#)
- [a everything Big CSV analyzer.thread num comments total skewed](#)
- [a everything Big CSV analyzer.thread num comments total](#) = thread_information['Thread num comments total']
- [a everything Big CSV analyzer.thread num comments tier 1](#) = thread_information['Thread num comments tier 1']
- [a everything Big CSV analyzer.thread num comments tier x](#) = thread_information['Thread num comments tier x']
- [a everything Big CSV analyzer.thread num questions total](#) = thread_information['Thread num questions total']
- [a everything Big CSV analyzer.thread num questions tier 1](#) = thread_information['Thread num questions tier 1']
- [a everything Big CSV analyzer.thread num questions tier x](#) = thread_information['Thread num questions tier x']
- [a everything Big CSV analyzer.thread num questions answered by iama host total](#)
- [a everything Big CSV analyzer.thread num questions answered by iama host tier 1](#)
- [a everything Big CSV analyzer.thread num questions answered by iama host tier x](#)
- [a everything Big CSV analyzer.thread num comments answered by iama host total](#)

- [a everything Big CSV analyzer.thread num comments answered by iama host tier 1](#)
- [a everything Big CSV analyzer.thread num comments answered by iama host tier x](#)
- [a everything Big CSV analyzer.thread average reaction time between comments total](#)
- [a everything Big CSV analyzer.thread average reaction time between comments tier 1](#)
- [a everything Big CSV analyzer.thread average reaction time between comments tier x](#)
- [a everything Big CSV analyzer.thread average reaction time between questions total](#)
- [a everything Big CSV analyzer.thread average reaction time between questions tier 1](#)
- [a everything Big CSV analyzer.thread average reaction time between questions tier x](#)
- [a everything Big CSV analyzer.thread average response to comment time iama host total](#)
- [a everything Big CSV analyzer.thread average response to comment time iama host tier 1](#)
- [a everything Big CSV analyzer.thread average response to comment time iama host tier x](#)
- [a everything Big CSV analyzer.thread average response to question time iama host total](#)
- [a everything Big CSV analyzer.thread average response to question time iama host tier 1](#)
- [a everything Big CSV analyzer.thread average response to question time iama host tier x](#)
- [a everything Big CSV analyzer.thread amount of questioners total](#)
- [a everything Big CSV analyzer.thread amount of questioners tier 1](#)
- [a everything Big CSV analyzer.thread amount of questioners tier x](#)
- [a everything Big CSV analyzer.thread amount of commentators total](#)
- [a everything Big CSV analyzer.thread amount of commentators tier 1](#)
- [a everything Big CSV analyzer.thread amount of commentators tier x](#)
- [a everything Big CSV analyzer.thread life span until last comment](#)
- [a everything Big CSV analyzer.thread life span until last question](#)

a_author_Information.py File Reference

Namespaces

- [a_author_Information](#)

Functions

- def [a_author_Information.check_script_arguments\(\)](#)
- def [a_author_Information.initialize_mongo_db_parameters\(\)](#)
- def [a_author_Information.write_csv_data\(\)](#)

Variables

- [a_author_Information.mongo_db_client_instance](#) = None
- [a_author_Information.mongo_db_author_instance](#) = None
- [a_author_Information.mongo_db_author_collection](#) = None
- int [a_author_Information.mongo_db_author_collection_original](#) = 0
- string [a_author_Information.argument_db_to_choose](#) = ""

a_iAMA_Commenttime_arr.py File Reference

Namespaces

- [a_iAMA_Commenttime_arr](#)

Functions

- def [a_iAMA_Commenttime_arr.check_script_arguments\(\)](#)
- def [a_iAMA_Commenttime_arr.initialize_mongo_db_parameters](#) (actually_processed_year)
- def [a_iAMA_Commenttime_arr.start_data_generation_for_analysis\(\)](#)
- def [a_iAMA_Commenttime_arr.prepare_data_for_graph\(\)](#)
- def [a_iAMA_Commenttime_arr.add_thread_list_to_global_list](#) (list_to_append)
- def [a_iAMA_Commenttime_arr.generate_data_to_be_analyzed\(\)](#)
- def [a_iAMA_Commenttime_arr.calculate_ar_mean_answer_time_for_questions](#) (id_of_thread, author_of_thread)
- def [a_iAMA_Commenttime_arr.check_if_comment_is_a_question](#) (given_string)
- def [a_iAMA_Commenttime_arr.check_if_comment_is_on_tier_1](#) (comment_parent_id)
- def [a_iAMA_Commenttime_arr.check_if_comment_is_not_from_thread_author](#) (author_of_thread, comment_author)
- def [a_iAMA_Commenttime_arr.check_if_comment_is_answer_from_thread_author](#) (author_of_thread, comment_actual_id, comments_cursor)
- def [a_iAMA_Commenttime_arr.calculate_time_difference](#) (comment_time_stamp, answer_time_stamp_iama_host)
- def [a_iAMA_Commenttime_arr.write_csv_data](#) (list_with_information)
- def [a_iAMA_Commenttime_arr.plot_generated_data\(\)](#)

Variables

- int [a_iAMA_Commenttime_arr.argument_year_beginning](#) = 0
- int [a_iAMA_Commenttime_arr.year_actually_in_progress](#) = 0
- int [a_iAMA_Commenttime_arr.argument_year_ending](#) = 0
- string [a_iAMA_Commenttime_arr.argument_tier_in_scope](#) = ""
- string [a_iAMA_Commenttime_arr.argument_plot_time_unit](#) = ""
- [a_iAMA_Commenttime_arr.mongo_DB_Client_Instance](#) = None
- [a_iAMA_Commenttime_arr.mongo_DB_Threads_Instance](#) = None
- [a_iAMA_Commenttime_arr.mongo_DB_Thread_Collection](#) = None
- [a_iAMA_Commenttime_arr.mongo_DB_Comments_Instance](#) = None
- list [a_iAMA_Commenttime_arr.list_To_Be_Plotted](#) = []
- list [a_iAMA_Commenttime_arr.global_thread_list](#) = []
- list [a_iAMA_Commenttime_arr.data_to_give_plotly](#) = []

a_iAMA_Commenttime_med.py File Reference

Namespaces

- [a iAMA Commenttime med](#)

Functions

- def [a iAMA Commenttime med.check script arguments](#) ()
- def [a iAMA Commenttime med.initialize mongo db parameters](#) (actually_processed_year)
- def [a iAMA Commenttime med.start data generation for analysis](#) ()
- def [a iAMA Commenttime med.prepare data for graph](#) ()
- def [a iAMA Commenttime med.add thread list to global list](#) (list_to_append)
- def [a iAMA Commenttime med.generate data to be analyzed](#) ()
- def [a iAMA Commenttime med.calculate ar mean answer time for questions](#) (id_of_thread, author_of_thread)
- def [a iAMA Commenttime med.check if comment is a question](#) (given_string)
- def [a iAMA Commenttime med.check if comment is on tier 1](#) (comment_parent_id)
- def [a iAMA Commenttime med.check if comment is not from thread author](#) (author_of_thread, comment_author)
- def [a iAMA Commenttime med.check if comment is answer from thread author](#) (author_of_thread, comment_actual_id, comments_cursor)
- def [a iAMA Commenttime med.calculate time difference](#) (comment_time_stamp, answer_time_stamp_iama_host)
- def [a iAMA Commenttime med.write csv data](#) (list_with_information)
- def [a iAMA Commenttime med.plot generated data](#) ()

Variables

- int [a iAMA Commenttime med.argument year beginning](#) = 0
- int [a iAMA Commenttime med.year actually in progress](#) = 0
- int [a iAMA Commenttime med.argument year ending](#) = 0
- string [a iAMA Commenttime med.argument tier in scope](#) = ""
- string [a iAMA Commenttime med.argument plot time unit](#) = ""
- [a iAMA Commenttime med.mongo DB Client Instance](#) = None
- [a iAMA Commenttime med.mongo DB Threads Instance](#) = None
- [a iAMA Commenttime med.mongo DB Thread Collection](#) = None
- [a iAMA Commenttime med.mongo DB Comments Instance](#) = None
- list [a iAMA Commenttime med.list To Be Plotted](#) = []
- list [a iAMA Commenttime med.global thread list](#) = []
- list [a iAMA Commenttime med.data to give plotly](#) = []

a_question_Answered_Yes_No_Extrema.py File Reference

Namespaces

- [a_question_Answered_Yes_No_Extrema](#)

Functions

- def [a_question_Answered_Yes_No_Extrema.check_script_arguments](#) ()
- def [a_question_Answered_Yes_No_Extrema.initialize_mongo_db_parameters](#) (actually_processed_year)
- def [a_question_Answered_Yes_No_Extrema.start_data_generation_for_analysis](#) ()
- def [a_question_Answered_Yes_No_Extrema.generate_data_now](#) ()
- def [a_question_Answered_Yes_No_Extrema.process_answered_questions_within_thread](#) (id_of_thread, author_of_thread, thread_creation_date)
- def [a_question_Answered_Yes_No_Extrema.check_if_comment_is_a_question](#) (given_string)
- def [a_question_Answered_Yes_No_Extrema.check_if_comment_is_not_from_thread_author](#) (author_of_thread, comment_author)
- def [a_question_Answered_Yes_No_Extrema.check_if_comment_has_been_answered_by_thread_author](#) (author_of_thread, comment_acutal_id, comments_cursor)
- def [a_question_Answered_Yes_No_Extrema.calculate_time_difference](#) (comment_time_stamp, answer_time_stamp_iama_host)
- def [a_question_Answered_Yes_No_Extrema.sort_questions](#) (list_which_is_to_be_sorted)
- def [a_question_Answered_Yes_No_Extrema.create_question_list_containing_all_years](#) (list_with_comments_per_years)
- def [a_question_Answered_Yes_No_Extrema.write_csv_and_count_unanswered](#) (list_with_comments)
- def [a_question_Answered_Yes_No_Extrema.plot_generated_data](#) ()

Variables

- int [a_question_Answered_Yes_No_Extrema.argument_year_beginning](#) = 0
- int [a_question_Answered_Yes_No_Extrema.year_actually_in_progress](#) = 0
- int [a_question_Answered_Yes_No_Extrema.argument_year_ending](#) = 0
- [a_question_Answered_Yes_No_Extrema.argument_sorting](#) = bool
- int [a_question_Answered_Yes_No_Extrema.argument_amount_of_top_quotes](#) = 0
- [a_question_Answered_Yes_No_Extrema.mongo_DB_Client_Instance](#) = None
- [a_question_Answered_Yes_No_Extrema.mongo_DB_Threads_Instance](#) = None
- [a_question_Answered_Yes_No_Extrema.mongo_DB_Thread_Collection](#) = None
- [a_question_Answered_Yes_No_Extrema.mongo_DB_Comments_Instance](#) = None
- list [a_question_Answered_Yes_No_Extrema.question_information_list](#) = []
- list [a_question_Answered_Yes_No_Extrema.data_to_give_plotly](#) = []

a_question_Answered_Yes_No_Tier_Percentage.py File Reference

Namespaces

- [a_question_Answered_Yes_No_Tier_Percentage](#)

Functions

- def [a_question_Answered_Yes_No_Tier_Percentage.check_script_arguments\(\)](#)
- def [a_question_Answered_Yes_No_Tier_Percentage.initialize_mongo_db_parameters\(actually_processed_year\)](#)
- def [a_question_Answered_Yes_No_Tier_Percentage.start_data_generation_for_analysis\(\)](#)
- def [a_question_Answered_Yes_No_Tier_Percentage.generate_data_to_be_analyzed\(\)](#)
- def [a_question_Answered_Yes_No_Tier_Percentage.question_answering_distribution_tier1_tierx_tierany\(id_of_thread, author_of_thread\)](#)
- def [a_question_Answered_Yes_No_Tier_Percentage.check_if_comment_is_a_question\(given_string\)](#)
- def [a_question_Answered_Yes_No_Tier_Percentage.check_if_comment_is_on_tier_1\(comment_parent_id\)](#)
- def [a_question_Answered_Yes_No_Tier_Percentage.check_if_comment_is_not_from_thread_author\(author_of_thread, comment_author\)](#)
- def [a_question_Answered_Yes_No_Tier_Percentage.check_if_comment_is_answer_from_thread_author\(author_of_thread, comment_actual_id, comments_cursor\)](#)
- def [a_question_Answered_Yes_No_Tier_Percentage.write_csv\(list_with_information\)](#)
- def [a_question_Answered_Yes_No_Tier_Percentage.add_local_list_to_global_list\(list_to_append\)](#)
- def [a_question_Answered_Yes_No_Tier_Percentage.prepare_data_for_graph\(\)](#)
- def [a_question_Answered_Yes_No_Tier_Percentage.plot_generated_data\(\)](#)

Variables

- int [a_question_Answered_Yes_No_Tier_Percentage.argument_year_beginning](#) = 0
- int [a_question_Answered_Yes_No_Tier_Percentage.year_actually_in_progress](#) = 0
- int [a_question_Answered_Yes_No_Tier_Percentage.argument_year_ending](#) = 0
- string [a_question_Answered_Yes_No_Tier_Percentage.argument_tier_in_scope](#) = ""
- [a_question_Answered_Yes_No_Tier_Percentage.mongo_DB_Client_Instance](#) = None
- [a_question_Answered_Yes_No_Tier_Percentage.mongo_DB_Threads_Instance](#) = None
- [a_question_Answered_Yes_No_Tier_Percentage.mongo_DB_Thread_Collection](#) = None
- [a_question_Answered_Yes_No_Tier_Percentage.mongo_DB_Comments_Instance](#) = None
- list [a_question_Answered_Yes_No_Tier_Percentage.global_question_list](#) = []
- list [a_question_Answered_Yes_No_Tier_Percentage.year_question_list](#) = []
- list [a_question_Answered_Yes_No_Tier_Percentage.data_to_give_plotly](#) = []

a_question_Tier_Distribution.py File Reference

Namespaces

- [a_question_Tier_Distribution](#)

Functions

- def [a_question_Tier_Distribution.initialize_mongo_db_parameters](#) (actually_processed_year)
- def [a_question_Tier_Distribution.check_script_arguments](#) ()
- def [a_question_Tier_Distribution.start_data_generation_for_analysis](#) ()
- def [a_question_Tier_Distribution.generate_data_to_be_analyzed](#) ()
- def [a_question_Tier_Distribution.question_distribution_tier1_tierx](#) (id_of_thread, author_of_thread)
- def [a_question_Tier_Distribution.check_if_comment_is_a_question](#) (given_string)
- def [a_question_Tier_Distribution.check_if_comment_is_on_tier_1](#) (comment_parent_id)
- def [a_question_Tier_Distribution.check_if_comment_is_not_from_thread_author](#) (author_of_thread, comment_author)
- def [a_question_Tier_Distribution.add_actual_year_list_to_global_list](#) (list_to_append)
- def [a_question_Tier_Distribution.write_csv](#) (list_with_information)
- def [a_question_Tier_Distribution.prepare_data_for_graph](#) ()
- def [a_question_Tier_Distribution.plot_generated_data](#) ()

Variables

- int [a_question_Tier_Distribution.argument_year_beginning](#) = 0
- int [a_question_Tier_Distribution.year_actually_in_progress](#) = 0
- int [a_question_Tier_Distribution.argument_year_ending](#) = 0
- [a_question_Tier_Distribution.mongo_DB_Client_Instance](#) = None
- [a_question_Tier_Distribution.mongo_DB_Threads_Instance](#) = None
- [a_question_Tier_Distribution.mongo_DB_Thread_Collection](#) = None
- [a_question_Tier_Distribution.mongo_DB_Comments_Instance](#) = None
- list [a_question_Tier_Distribution.current_year_question_list](#) = []
- list [a_question_Tier_Distribution.global_year_question_list](#) = []
- list [a_question_Tier_Distribution.data_to_give_plotly](#) = []

a_thread_Lifespan_N_Average_Commenttime.py File Reference

Namespaces

- [a_thread_Lifespan_N_Average_Commenttime](#)

Functions

- def [a_thread_Lifespan_N_Average_Commenttime.check_script_arguments](#) ()
- def [a_thread_Lifespan_N_Average_Commenttime.initialize_mongo_db_parameters](#) (actually_processed_year)
- def [a_thread_Lifespan_N_Average_Commenttime.start_data_generation_for_analysis](#) ()
- def [a_thread_Lifespan_N_Average_Commenttime.prepare_data_for_graph_life_span](#) ()
- def [a_thread_Lifespan_N_Average_Commenttime.prepare_data_for_comment_time](#) ()
- def [a_thread_Lifespan_N_Average_Commenttime.generate_data_to_be_analyzed](#) ()
- def [a_thread_Lifespan_N_Average_Commenttime.calculate_time_difference](#) (id_of_thread, creation_date_of_thread)
- def [a_thread_Lifespan_N_Average_Commenttime.write_csv](#) (list_with_information)
- def [a_thread_Lifespan_N_Average_Commenttime.add_thread_list_to_global_list](#) (list_to_append)
- def [a_thread_Lifespan_N_Average_Commenttime.prepare_dict_by_time_separation_for_comment_time](#) ()
- def [a_thread_Lifespan_N_Average_Commenttime.plot_generated_data](#) ()

Variables

- int [a_thread_Lifespan_N_Average_Commenttime.argument_year_beginning](#) = 0
- string [a_thread_Lifespan_N_Average_Commenttime.argument_calculation](#) = ""
- int [a_thread_Lifespan_N_Average_Commenttime.argument_year_ending](#) = 0
- int [a_thread_Lifespan_N_Average_Commenttime.year_actually_in_progress](#) = 0
- string [a_thread_Lifespan_N_Average_Commenttime.argument_plot_time_unit](#) = ""
- [a_thread_Lifespan_N_Average_Commenttime.mongo_DB_Client_Instance](#) = None
- [a_thread_Lifespan_N_Average_Commenttime.mongo_DB_Threads_Instance](#) = None
- [a_thread_Lifespan_N_Average_Commenttime.mongo_DB_Thread_Collection](#) = None
- [a_thread_Lifespan_N_Average_Commenttime.mongo_DB_Comments_Instance](#) = None
- list [a_thread_Lifespan_N_Average_Commenttime.global_thread_list](#) = []
- list [a_thread_Lifespan_N_Average_Commenttime.temp_time_difference_list](#) = []
- list [a_thread_Lifespan_N_Average_Commenttime.list_with_currents_year_infos](#) = []
- list [a_thread_Lifespan_N_Average_Commenttime.data_to_give_plotly](#) = []

a_thread_Lifespan_N_Average_Questiontime.py File Reference

Namespaces

- [a_thread_Lifespan_N_Average_Questiontime](#)

Functions

- def [a_thread_Lifespan_N_Average_Questiontime.check_script_arguments\(\)](#)
- def [a_thread_Lifespan_N_Average_Questiontime.initialize_mongo_db_parameters](#) (actually_processed_year)
- def [a_thread_Lifespan_N_Average_Questiontime.start_data_generation_for_analysis\(\)](#)
- def [a_thread_Lifespan_N_Average_Questiontime.prepare_data_for_graph_life_span\(\)](#)
- def [a_thread_Lifespan_N_Average_Questiontime.prepare_data_for_comment_time\(\)](#)
- def [a_thread_Lifespan_N_Average_Questiontime.generate_data_to_be_analyzed\(\)](#)
- def [a_thread_Lifespan_N_Average_Questiontime.calculate_time_difference](#) (id_of_thread, creation_date_of_thread)
- def [a_thread_Lifespan_N_Average_Questiontime.write_csv](#) (list_with_information)
- def [a_thread_Lifespan_N_Average_Questiontime.add_thread_list_to_global_list](#) (list_to_append)
- def [a_thread_Lifespan_N_Average_Questiontime.prepare_dict_by_time_separation_for_comment_time\(\)](#)
- def [a_thread_Lifespan_N_Average_Questiontime.plot_generated_data\(\)](#)

Variables

- int [a_thread_Lifespan_N_Average_Questiontime.argument_year_beginning](#) = 0
- string [a_thread_Lifespan_N_Average_Questiontime.argument_calculation](#) = ""
- int [a_thread_Lifespan_N_Average_Questiontime.argument_year_ending](#) = 0
- int [a_thread_Lifespan_N_Average_Questiontime.year_actually_in_progress](#) = 0
- string [a_thread_Lifespan_N_Average_Questiontime.argument_plot_time_unit](#) = ""
- [a_thread_Lifespan_N_Average_Questiontime.mongo_DB_Client_Instance](#) = None
- [a_thread_Lifespan_N_Average_Questiontime.mongo_DB_Threads_Instance](#) = None
- [a_thread_Lifespan_N_Average_Questiontime.mongo_DB_Thread_Collection](#) = None
- [a_thread_Lifespan_N_Average_Questiontime.mongo_DB_Comments_Instance](#) = None
- list [a_thread_Lifespan_N_Average_Questiontime.global_thread_list](#) = []
- list [a_thread_Lifespan_N_Average_Questiontime.temp_time_difference_list](#) = []
- list [a_thread_Lifespan_N_Average_Questiontime.list_with_currents_year_infos](#) = []
- list [a_thread_Lifespan_N_Average_Questiontime.data_to_give_plotly](#) = []

ar_analyzer.py File Reference

Namespaces

- [ar_analyzer](#)

Functions

- [def ar_analyzer.average_means_of_values_f_threads \(\)](#)
- [def ar_analyzer.std_derivation \(\)](#)

Variables

- [ar_analyzer.thread_information](#)
- [ar_analyzer.thread_year](#) = thread_information['Year']
- [ar_analyzer.thread_id](#) = thread_information['Thread id']
- [ar_analyzer.thread_author](#) = thread_information['Thread author']
- [ar_analyzer.thread_ups](#) = thread_information['Thread ups']
- [ar_analyzer.thread_downs](#) = thread_information['Thread downs']
- [ar_analyzer.thread_creation_time_stamp](#) = thread_information['Thread creation time stamp']
- [ar_analyzer.thread_average_comment_vote_score_total](#)
- [ar_analyzer.thread_average_comment_vote_score_tier_1](#)
- [ar_analyzer.thread_average_comment_vote_score_tier_x](#)
- [ar_analyzer.thread_average_question_vote_score_total](#)
- [ar_analyzer.thread_average_question_vote_score_tier_1](#)
- [ar_analyzer.thread_average_question_vote_score_tier_x](#)
- [ar_analyzer.thread_num_comments_total_skewed](#)
- [ar_analyzer.thread_num_comments_total](#) = thread_information['Thread num comments total']
- [ar_analyzer.thread_num_comments_tier_1](#) = thread_information['Thread num comments tier 1']
- [ar_analyzer.thread_num_comments_tier_x](#) = thread_information['Thread num comments tier x']
- [ar_analyzer.thread_num_questions_total](#) = thread_information['Thread num questions total']
- [ar_analyzer.thread_num_questions_tier_1](#) = thread_information['Thread num questions tier 1']
- [ar_analyzer.thread_num_questions_tier_x](#) = thread_information['Thread num questions tier x']
- [ar_analyzer.thread_num_questions_answered_by_iamahost_total](#)
- [ar_analyzer.thread_num_questions_answered_by_iamahost_tier_1](#)
- [ar_analyzer.thread_num_questions_answered_by_iamahost_tier_x](#)
- [ar_analyzer.thread_num_comments_answered_by_iamahost_total](#)
- [ar_analyzer.thread_num_comments_answered_by_iamahost_tier_1](#)
- [ar_analyzer.thread_num_comments_answered_by_iamahost_tier_x](#)
- [ar_analyzer.thread_average_reaction_time_between_comments_total](#)
- [ar_analyzer.thread_average_reaction_time_between_comments_tier_1](#)
- [ar_analyzer.thread_average_reaction_time_between_comments_tier_x](#)
- [ar_analyzer.thread_average_reaction_time_between_questions_total](#)
- [ar_analyzer.thread_average_reaction_time_between_questions_tier_1](#)
- [ar_analyzer.thread_average_reaction_time_between_questions_tier_x](#)
- [ar_analyzer.thread_average_response_to_comment_time_iamahost_total](#)
- [ar_analyzer.thread_average_response_to_comment_time_iamahost_tier_1](#)
- [ar_analyzer.thread_average_response_to_comment_time_iamahost_tier_x](#)
- [ar_analyzer.thread_average_response_to_question_time_iamahost_total](#)
- [ar_analyzer.thread_average_response_to_question_time_iamahost_tier_1](#)
- [ar_analyzer.thread_average_response_to_question_time_iamahost_tier_x](#)
- [ar_analyzer.thread_amount_of_questioners_total](#)
- [ar_analyzer.thread_amount_of_questioners_tier_1](#)
- [ar_analyzer.thread_amount_of_questioners_tier_x](#)

- [ar analyzer.thread amount of commentators total](#)
- [ar analyzer.thread amount of commentators tier 1](#)
- [ar analyzer.thread amount of commentators tier x](#)
- [ar analyzer.thread life span until last comment](#)
- [ar analyzer.thread life span until last question](#)

c_crawl_Author_Information.py File Reference

Namespaces

- [c_crawl_Author_Information](#)

Functions

- def [c_crawl_Author_Information.check_script_arguments](#) ()
- def [c_crawl_Author_Information.initialize_mongo_db_parameters](#) (actually_processed_year)
- def [c_crawl_Author_Information.start_data_generation_for_analysis](#) ()
- def [c_crawl_Author_Information.generate_data_now](#) ()
- def [c_crawl_Author_Information.calculate_time_difference](#) (time_value_1, time_value_2)
- def [c_crawl_Author_Information.get_author_information](#) (name_of_author)

Variables

- int [c_crawl_Author_Information.argument_year_beginning](#) = 0
- int [c_crawl_Author_Information.year_actually_in_progress](#) = 0
- int [c_crawl_Author_Information.argument_year_ending](#) = 0
- string [c_crawl_Author_Information.argument_inverse_crawling](#) = ""
- [c_crawl_Author_Information.mongo_db_client_instance](#) = None
- [c_crawl_Author_Information.mongo_db_threads_instance](#) = None
- [c_crawl_Author_Information.mongo_db_thread_collection](#) = None
- [c_crawl_Author_Information.mongo_db_author_instance](#) = None
- [c_crawl_Author_Information.reddit_instance](#) =
praw.Reddit(user_agent="University_Regensburg_iAMA_Crawler_0.001")

c_crawl_Differences.py File Reference

Namespaces

- [c_crawl_Differences](#)

Functions

- [def c_crawl_Differences.check_script_arguments \(\)](#)
- [def c_crawl_Differences.initialize_mongo_db_parameters \(\)](#)
- [def c_crawl_Differences.crawl_missing_collection_into_comments_database \(name_of_missing_collection\)](#)
- [def c_crawl_Differences.check_if_collection_is_missing_in_comments_database \(\)](#)
- [def c_crawl_Differences.crawl_missing_collection_into_threads_database \(name_of_missing_collection\)](#)
- [def c_crawl_Differences.check_if_collection_is_missing_in_threads_database \(\)](#)
- [def c_crawl_Differences.start_crawling_for_diffs \(\)](#)

Variables

- [c_crawl_Differences.mongo_DB_Client_Instance = None](#)
- [c_crawl_Differences.mongo_DB_Threads_Instance = None](#)
- [c_crawl_Differences.mongo_DB_Thread_Collection = None](#)
- [c_crawl_Differences.mongo_DB_Comments_Instance = None](#)
- [c_crawl_Differences.mongo_DB_Comments_Collection = None](#)
- [string c_crawl_Differences.argument_year_beginning = ""](#)
- [string c_crawl_Differences.argument_year_ending = ""](#)
- [string c_crawl_Differences.argument_inverse_crawling = ""](#)

c_crawl_Random_Author_Information.py File Reference

Namespaces

- [c_crawl_Random_Author_Information](#)

Functions

- def [c_crawl_Random_Author_Information.check_script_arguments](#) ()
- def [c_crawl_Random_Author_Information.initialize_mongo_db_parameters](#) ()
- def [c_crawl_Random_Author_Information.start_data_generation_for_analysis](#) ()
- def [c_crawl_Random_Author_Information.generate_data_now](#) (randomized_author_name)
- def [c_crawl_Random_Author_Information.calculate_time_difference](#) (time_value_1, time_value_2)
- def [c_crawl_Random_Author_Information.get_author_information](#) (name_of_author)

Variables

- [c_crawl_Random_Author_Information.argument_limit_crawling_amount](#) = None
- [c_crawl_Random_Author_Information.mongo_db_client_instance](#) = None
- [c_crawl_Random_Author_Information.mongo_db_random_author_instance](#) = None
- [c_crawl_Random_Author_Information.mongo_db_random_author_collection](#) = None
- [c_crawl_Random_Author_Information.mongo_db_iama_author_instance](#) = None
- [c_crawl_Random_Author_Information.mongo_db_iama_author_collection](#) = None
- int [c_crawl_Random_Author_Information.mongo_db_iama_author_collection_amount](#) = 0
- [c_crawl_Random_Author_Information.reddit_instance](#) =
praw.Reddit(user_agent="University_Regensburg_iAMA_Crawler_0.001")

c_crawl_Threads_N_Comments.py File Reference

Namespaces

- [c_crawl_Threads_N_Comments](#)

Functions

- def [c_crawl_Threads_N_Comments.initialize_mongo_db_parameters](#) ()
- def [c_crawl_Threads_N_Comments.check_script_arguments](#) ()
- def [c_crawl_Threads_N_Comments.convert_argument_year_to_epoch](#) (year)
- def [c_crawl_Threads_N_Comments.crawl_data](#) ()
- def [c_crawl_Threads_N_Comments.crawl_threads](#) ()
- def [c_crawl_Threads_N_Comments.crawl_comments](#) ()
- def [c_crawl_Threads_N_Comments.check_if_coll_in_db_already_exists_up2date](#) (submission)

Variables

- [c_crawl_Threads_N_Comments.mongo_DB_Client_Instance](#) = None
- [c_crawl_Threads_N_Comments.reddit_Instance](#) = None
- [c_crawl_Threads_N_Comments.argument_crawl_type](#) = None
- [c_crawl_Threads_N_Comments.argument_year_beginning](#) = None
- [c_crawl_Threads_N_Comments.argument_year_end](#) = None
- [c_crawl_Threads_N_Comments.argument_hours_to_shift](#) = None
- [c_crawl_Threads_N_Comments.time_shift_difference](#)

d_create_Big_CSV_ar.py File Reference

Namespaces

- [d_create_Big_CSV_ar](#)

Functions

- def [d_create_Big_CSV_ar.check_script_arguments](#) ()
- def [d_create_Big_CSV_ar.initialize_mongo_db_parameters](#) (actually_processed_year)
- def [d_create_Big_CSV_ar.start_data_generation_for_analysis](#) ()
- def [d_create_Big_CSV_ar.generate_data](#) ()
- def [d_create_Big_CSV_ar.process_specific_thread](#) (thread_id, thread_creation_time_stamp, thread_author)
- def [d_create_Big_CSV_ar.check_if_comment_is_a_question](#) (given_string)
- def [d_create_Big_CSV_ar.check_if_comment_is_on_tier_1](#) (comment_parent_id)
- def [d_create_Big_CSV_ar.check_if_comment_is_not_from_thread_author](#) (author_of_thread, comment_author)
- def [d_create_Big_CSV_ar.check_if_comment_has_been_answered_by_thread_author](#) (author_of_thread, comment_actual_id, comments_cursor)
- def [d_create_Big_CSV_ar.calculate_time_difference](#) (comment_time_stamp, answer_time_stamp_iama_host)
- def [d_create_Big_CSV_ar.calculate_reaction_time_average](#) (list_to_be_processed, thread_creation_time_stamp)
- def [d_create_Big_CSV_ar.calculate_life_span](#) (thread_creation_time_stamp, time_value_of_last_comment, time_value_of_last_question)
- def [d_create_Big_CSV_ar.add_actual_year_list_to_global_list](#) (list_to_append)
- def [d_create_Big_CSV_ar.write_csv_data](#) (list_with_information)

Variables

- int [d_create_Big_CSV_ar.argument_year_beginning](#) = 0
- int [d_create_Big_CSV_ar.argument_year_ending](#) = 0
- int [d_create_Big_CSV_ar.year_actually_in_progress](#) = 0
- list [d_create_Big_CSV_ar.list_current_year](#) = []
- list [d_create_Big_CSV_ar.list_global_year](#) = []

d_create_Big_CSV_med.py File Reference

Namespaces

- [d_create_Big_CSV_med](#)

Functions

- def [d_create_Big_CSV_med.check_script_arguments](#) ()
- def [d_create_Big_CSV_med.initialize_mongo_db_parameters](#) (actually_processed_year)
- def [d_create_Big_CSV_med.start_data_generation_for_analysis](#) ()
- def [d_create_Big_CSV_med.generate_data](#) ()
- def [d_create_Big_CSV_med.process_specific_thread](#) (thread_id, thread_creation_time_stamp, thread_author)
- def [d_create_Big_CSV_med.check_if_comment_is_a_question](#) (given_string)
- def [d_create_Big_CSV_med.check_if_comment_is_on_tier_1](#) (comment_parent_id)
- def [d_create_Big_CSV_med.check_if_comment_is_not_from_thread_author](#) (author_of_thread, comment_author)
- def [d_create_Big_CSV_med.check_if_comment_has_been_answered_by_thread_author](#) (author_of_thread, comment_actual_id, comments_cursor)
- def [d_create_Big_CSV_med.calculate_time_difference](#) (comment_time_stamp, answer_time_stamp_iama_host)
- def [d_create_Big_CSV_med.calculate_reaction_time_average](#) (list_to_be_processed, thread_creation_time_stamp)
- def [d_create_Big_CSV_med.calculate_life_span](#) (thread_creation_time_stamp, time_value_of_last_comment, time_value_of_last_question)
- def [d_create_Big_CSV_med.add_actual_year_list_to_global_list](#) (list_to_append)
- def [d_create_Big_CSV_med.write_csv_data](#) (list_with_information)

Variables

- int [d_create_Big_CSV_med.argument_year_beginning](#) = 0
- int [d_create_Big_CSV_med.argument_year_ending](#) = 0
- int [d_create_Big_CSV_med.year_actually_in_progress](#) = 0
- list [d_create_Big_CSV_med.list_current_year](#) = []
- list [d_create_Big_CSV_med.list_global_year](#) = []

PlotlyBarChart.py File Reference

Classes

- class [PlotlyBarChart.PlotlyBarChart](#)

Namespaces

- [PlotlyBarChart](#)

PlotlyBarChart_5_Bars.py File Reference

Classes

- class [PlotlyBarChart_5_Bars.PlotlyBarChart5Bars](#)

Namespaces

- [PlotlyBarChart_5_Bars](#)

Index

__init__
PlotlyBarChart::PlotlyBarChart 104
PlotlyBarChart_5_Bars::PlotlyBarChart5Bars 109
a__everything_Big_CSV_analyzer 5
arr_of_values_f_authors 8
author_amount_creation_iama_threads 16
author_amount_creation_iama_threads_randomize
d 16
author_amount_creation_other_threads 16
author_amount_creation_other_threads_randomize
d 16
author_amount_of_comments_except_iama 16
author_amount_of_comments_except_iama_rando
mized 16
author_amount_of_comments_iama 16
author_amount_of_comments_iama_randomized
17
author_author_birth_date 17
author_author_birth_date_randomized 17
author_author_comment_karma_amount 17
author_author_comment_karma_amount_randomiz
ed 17
author_author_link_karma_amount 17
author_author_link_karma_amount_randomized
17
author_author_name 17
author_author_name_randomized 17
author_comment_creation_every_x_sec 17
author_comment_creation_every_x_sec_randomize
d 18
author_information_iama 18
author_information_iama_sampleset_randomized
18
author_information_random 18
author_information_random_sampleset_randomize
d 18
author_thread_creation_every_x_sec 18
author_thread_creation_every_x_sec_randomized
18
author_time_acc_birth_first_iama_thread 18
author_time_acc_birth_first_iama_thread_rando
mized 18
author_time_diff_acc_creation_n_first_comment
19
author_time_diff_acc_creation_n_first_comment_r
andomized 19
author_time_diff_acc_creation_n_first_thread 19
author_time_diff_acc_creation_n_first_thread_rand
omized 19
average_means_of_values_f_authors 8
average_means_of_values_f_threads 8
calculate_t_tests_of_author_values 8
inplace 19
median_of_values_f_authors 9
median_of_values_f_authors_randomized 9
question_answered_by_iAMA_host 19
question_information 19
question_overall_correlation 9
question_ups 19
random_author_amount_creation_iama_threads
19
random_author_amount_creation_iama_threads_ra
ndomized 19
random_author_amount_creation_other_threads
20
random_author_amount_creation_other_threads_ra
ndomized 20
random_author_amount_of_comments_except_iam
a 20
random_author_amount_of_comments_except_iam
a_randomized 20
random_author_amount_of_comments_iama 20
random_author_amount_of_comments_iama_rand
omized 20
random_author_author_birth_date 20
random_author_author_birth_date_randomized
20
random_author_author_comment_karma_amount
20
random_author_author_comment_karma_amount_
randomized 21
random_author_author_link_karma_amount 21
random_author_author_link_karma_amount_rando
mized 21
random_author_author_name 21
random_author_author_name_randomized 21
random_author_comment_creation_every_x_sec
21
random_author_comment_creation_every_x_sec_r
andomized 21
random_author_thread_creation_every_x_sec 21
random_author_thread_creation_every_x_sec_rand
omized 21
random_author_time_acc_birth_first_iama_thread
22
random_author_time_acc_birth_first_iama_thread_
randomized 22
random_author_time_diff_acc_creation_n_first_co
mment 22
random_author_time_diff_acc_creation_n_first_co
mment_randomized 22
random_author_time_diff_acc_creation_n_first_thr
ead 22
random_author_time_diff_acc_creation_n_first_thr
ead_randomized 22

relation_thread_amount_of_commentators_total_and_num_comments_answered_by_iama_host 9
 relation_question_upvotes_with_amount_of_questions_answered_by_iama_host 9
 relation_thread_amount_of_questioners_total_and_num_questions_answered_by_iama_host 9
 relation_thread_amount_of_questions_and_amount_questions_answered_by_iama_host 10
 relation_thread_downvotes_and_iama_host_response_time_comments 10
 relation_thread_downvotes_and_iama_host_response_time_questions 10
 relation_thread_downvotes_with_amount_of_comments 10
 relation_thread_downvotes_with_amount_of_questions 11
 relation_thread_lifespan_to_last_comment_and_amount_of_comments 11
 relation_thread_lifespan_to_last_comment_and_amount_of_questions 11
 relation_thread_lifespan_to_last_comment_and_iama_host_response_time_to_comments 11
 relation_thread_lifespan_to_last_comment_and_iama_host_response_time_to_questions 11
 relation_thread_lifespan_to_last_question_and_amount_of_comments 12
 relation_thread_lifespan_to_last_question_and_amount_of_question 12
 relation_thread_lifespan_to_last_question_and_iama_host_response_time_to_comments 12
 relation_thread_lifespan_to_last_question_and_iama_host_response_time_to_questions 12
 relation_thread_reaction_time_comments_and_amount_of_comments_the_iama_host_answered_to 13
 relation_thread_reaction_time_comments_and_amount_of_questions_the_iama_host_answered_to 13
 relation_thread_reaction_time_comments_and_iama_host_response_time_to_comments 13
 relation_thread_reaction_time_comments_and_iama_host_response_time_to_questions 13
 relation_thread_reaction_time_questions_and_amount_of_comments_the_iama_host_answered_to 14
 relation_thread_reaction_time_questions_and_amount_of_questions_the_iama_host_answered_to 14
 relation_thread_reaction_time_questions_and_iama_host_response_time_to_comments 14
 relation_thread_reaction_time_questions_and_iama_host_response_time_to_questions 14
 relation_thread_upvotes_and_iama_host_response_time_comments 15
 relation_thread_upvotes_and_iama_host_response_time_questions 15

relation_thread_upvotes_with_amount_of_comments 15
 relation_thread_upvotes_with_amount_of_questions 15
 thread_amount_of_commentators_tier_1 22
 thread_amount_of_commentators_tier_x 22
 thread_amount_of_commentators_total 22
 thread_amount_of_questioners_tier_1 22
 thread_amount_of_questioners_tier_x 23
 thread_amount_of_questioners_total 23
 thread_author 23
 thread_average_comment_vote_score_tier_1 23
 thread_average_comment_vote_score_tier_x 23
 thread_average_comment_vote_score_total 23
 thread_average_question_vote_score_tier_1 23
 thread_average_question_vote_score_tier_x 23
 thread_average_question_vote_score_total 23
 thread_average_reaction_time_between_comments_tier_1 23
 thread_average_reaction_time_between_comments_tier_x 23
 thread_average_reaction_time_between_comments_total 24
 thread_average_reaction_time_between_questions_tier_1 24
 thread_average_reaction_time_between_questions_tier_x 24
 thread_average_reaction_time_between_questions_total 24
 thread_average_response_to_comment_time_iama_host_tier_1 24
 thread_average_response_to_comment_time_iama_host_tier_x 24
 thread_average_response_to_comment_time_iama_host_total 24
 thread_average_response_to_question_time_iama_host_tier_1 24
 thread_average_response_to_question_time_iama_host_tier_x 24
 thread_average_response_to_question_time_iama_host_total 24
 thread_creation_time_stamp 24
 thread_downs 25
 thread_id 25
 thread_information 25
 thread_life_span_until_last_comment 25
 thread_life_span_until_last_question 25
 thread_num_comments_answered_by_iama_host_tier_1 25
 thread_num_comments_answered_by_iama_host_tier_x 25
 thread_num_comments_answered_by_iama_host_total 25
 thread_num_comments_tier_1 25
 thread_num_comments_tier_x 25
 thread_num_comments_total 26
 thread_num_comments_total_skewed 26

thread_num_questions_answered_by_iama_host_tier_1 26
 thread_num_questions_answered_by_iama_host_tier_x 26
 thread_num_questions_answered_by_iama_host_total 26
 thread_num_questions_tier_1 26
 thread_num_questions_tier_x 26
 thread_num_questions_total 26
 thread_overall_correlation 16
 thread_ups 26
 thread_year 26
 a__everything_Big_CSV_analyzer.py 115
 a_author_Information 27
 argument_db_to_choose 28
 check_script_arguments 27
 initialize_mongo_db_parameters 27
 mongo_db_author_collection 28
 mongo_db_author_collection_original 28
 mongo_db_author_instance 28
 mongo_db_client_instance 28
 write_csv_data 27
 a_author_Information.py 120
 a_iAMA_Commenttime_arr 29
 add_thread_list_to_global_list 29
 argument_plot_time_unit 33
 argument_tier_in_scope 33
 argument_year_beginning 33
 argument_year_ending 33
 calculate_ar_mean_answer_time_for_questions 30
 calculate_time_difference 30
 check_if_comment_is_a_question 30
 check_if_comment_is_answer_from_thread_author 31
 check_if_comment_is_not_from_thread_author 31
 check_if_comment_is_on_tier_1 31
 check_script_arguments 31
 data_to_give_plotly 34
 generate_data_to_be_analyzed 32
 global_thread_list 34
 initialize_mongo_db_parameters 32
 list_To_Be_Plotted 34
 mongo_DB_Client_Instance 34
 mongo_DB_Comments_Instance 34
 mongo_DB_Thread_Collection 34
 mongo_DB_Threads_Instance 34
 plot_generated_data 32
 prepare_data_for_graph 32
 start_data_generation_for_analysis 33
 write_csv_data 33
 year_actually_in_progress 34
 a_iAMA_Commenttime_arr.py 121
 a_iAMA_Commenttime_med 35
 add_thread_list_to_global_list 35
 argument_plot_time_unit 39
 argument_tier_in_scope 39
 argument_year_beginning 39
 argument_year_ending 39
 calculate_ar_mean_answer_time_for_questions 36
 calculate_time_difference 36
 check_if_comment_is_a_question 36
 check_if_comment_is_answer_from_thread_author 37
 check_if_comment_is_not_from_thread_author 37
 check_if_comment_is_on_tier_1 37
 check_script_arguments 37
 data_to_give_plotly 40
 generate_data_to_be_analyzed 38
 global_thread_list 40
 initialize_mongo_db_parameters 38
 list_To_Be_Plotted 40
 mongo_DB_Client_Instance 40
 mongo_DB_Comments_Instance 40
 mongo_DB_Thread_Collection 40
 mongo_DB_Threads_Instance 40
 plot_generated_data 38
 prepare_data_for_graph 38
 start_data_generation_for_analysis 39
 write_csv_data 39
 year_actually_in_progress 40
 a_iAMA_Commenttime_med.py 122
 a_question_Answered_Yes_No_Extrema 41
 argument_amount_of_top_quotes 45
 argument_sorting 45
 argument_year_beginning 45
 argument_year_ending 45
 calculate_time_difference 41
 check_if_comment_has_been_answered_by_thread_author 42
 check_if_comment_is_a_question 42
 check_if_comment_is_not_from_thread_author 42
 check_script_arguments 43
 create_question_list_containing_all_years 43
 data_to_give_plotly 45
 generate_data_now 43
 initialize_mongo_db_parameters 43
 mongo_DB_Client_Instance 46
 mongo_DB_Comments_Instance 46
 mongo_DB_Thread_Collection 46
 mongo_DB_Threads_Instance 46
 plot_generated_data 44
 process_answered_questions_within_thread 44
 question_information_list 46
 sort_questions 44
 start_data_generation_for_analysis 44
 write_csv_and_count_unanswered 45
 year_actually_in_progress 46
 a_question_Answered_Yes_No_Extrema.py 123
 a_question_Answered_Yes_No_Tier_Percentage 47

add_local_list_to_global_list 47
 argument_tier_in_scope 51
 argument_year_beginning 51
 argument_year_ending 51
 check_if_comment_is_a_question 48
 check_if_comment_is_answer_from_thread_author 48
 check_if_comment_is_not_from_thread_author 48
 check_if_comment_is_on_tier_1 49
 check_script_arguments 49
 data_to_give_plotly 51
 generate_data_to_be_analyzed 49
 global_question_list 51
 initialize_mongo_db_parameters 49
 mongo_DB_Client_Instance 51
 mongo_DB_Comments_Instance 51
 mongo_DB_Thread_Collection 52
 mongo_DB_Threads_Instance 52
 plot_generated_data 50
 prepare_data_for_graph 50
 question_answering_distribution_tier1_tierx_tieran_y 50
 start_data_generation_for_analysis 50
 write_csv 51
 year_actually_in_progress 52
 year_question_list 52
 a_question_Answered_Yes_No_Tier_Percentage.py 124
 a_question_Tier_Distribution 53
 add_actual_year_list_to_global_list 53
 argument_year_beginning 56
 argument_year_ending 56
 check_if_comment_is_a_question 53
 check_if_comment_is_not_from_thread_author 54
 check_if_comment_is_on_tier_1 54
 check_script_arguments 54
 current_year_question_list 57
 data_to_give_plotly 57
 generate_data_to_be_analyzed 54
 global_year_question_list 57
 initialize_mongo_db_parameters 55
 mongo_DB_Client_Instance 57
 mongo_DB_Comments_Instance 57
 mongo_DB_Thread_Collection 57
 mongo_DB_Threads_Instance 57
 plot_generated_data 55
 prepare_data_for_graph 55
 question_distribution_tier1_tierx 55
 start_data_generation_for_analysis 56
 write_csv 56
 year_actually_in_progress 57
 a_question_Tier_Distribution.py 125
 a_thread_Lifespan_N_Average_Commenttime 58
 add_thread_list_to_global_list 58
 argument_calculation 61
 argument_plot_time_unit 61
 argument_year_beginning 61
 argument_year_ending 61
 calculate_time_difference 59
 check_script_arguments 59
 data_to_give_plotly 61
 generate_data_to_be_analyzed 59
 global_thread_list 62
 initialize_mongo_db_parameters 59
 list_with_currents_year_infos 62
 mongo_DB_Client_Instance 62
 mongo_DB_Comments_Instance 62
 mongo_DB_Thread_Collection 62
 mongo_DB_Threads_Instance 62
 plot_generated_data 60
 prepare_data_for_comment_time 60
 prepare_data_for_graph_life_span 60
 prepare_dict_by_time_separation_for_comment_time 60
 start_data_generation_for_analysis 61
 temp_time_difference_list 62
 write_csv 61
 year_actually_in_progress 62
 a_thread_Lifespan_N_Average_Commenttime.py 126
 a_thread_Lifespan_N_Average_Questiontime 63
 add_thread_list_to_global_list 63
 argument_calculation 66
 argument_plot_time_unit 66
 argument_year_beginning 66
 argument_year_ending 66
 calculate_time_difference 63
 check_script_arguments 64
 data_to_give_plotly 66
 generate_data_to_be_analyzed 64
 global_thread_list 67
 initialize_mongo_db_parameters 64
 list_with_currents_year_infos 67
 mongo_DB_Client_Instance 67
 mongo_DB_Comments_Instance 67
 mongo_DB_Thread_Collection 67
 mongo_DB_Threads_Instance 67
 plot_generated_data 65
 prepare_data_for_comment_time 65
 prepare_data_for_graph_life_span 65
 prepare_dict_by_time_separation_for_comment_time 65
 start_data_generation_for_analysis 66
 temp_time_difference_list 67
 write_csv 66
 year_actually_in_progress 67
 a_thread_Lifespan_N_Average_Questiontime.py 127
 add_actual_year_list_to_global_list
 a_question_Tier_Distribution 53
 d_create_Big_CSV_ar 90
 d_create_Big_CSV_med 96

add_local_list_to_global_list
 a_question_Answered_Yes_No_Tier_Percentage 47
 add_thread_list_to_global_list
 a_iAMA_Commenttime_arr 29
 a_iAMA_Commenttime_med 35
 a_thread_Lifespan_N_Average_Commenttime 58
 a_thread_Lifespan_N_Average_Questiontime 63
 annotations_1
 PlotlyBarChart_5_Bars::PlotlyBarChart5Bars 111
 annotations_2
 PlotlyBarChart_5_Bars::PlotlyBarChart5Bars 111
 annotations_3
 PlotlyBarChart_5_Bars::PlotlyBarChart5Bars 111
 annotations_4
 PlotlyBarChart_5_Bars::PlotlyBarChart5Bars 111
 annotations_5
 PlotlyBarChart_5_Bars::PlotlyBarChart5Bars 111
 annotations_all
 PlotlyBarChart_5_Bars::PlotlyBarChart5Bars 111
 ar_analyzer 68
 average_means_of_values_f_threads 69
 std_derivation 69
 thread_amount_of_commentators_tier_1 69
 thread_amount_of_commentators_tier_x 69
 thread_amount_of_commentators_total 69
 thread_amount_of_questioners_tier_1 69
 thread_amount_of_questioners_tier_x 69
 thread_amount_of_questioners_total 69
 thread_author 70
 thread_average_comment_vote_score_tier_1 70
 thread_average_comment_vote_score_tier_x 70
 thread_average_comment_vote_score_total 70
 thread_average_question_vote_score_tier_1 70
 thread_average_question_vote_score_tier_x 70
 thread_average_question_vote_score_total 70
 thread_average_reaction_time_between_comments_tier_1 70
 thread_average_reaction_time_between_comments_tier_x 70
 thread_average_reaction_time_between_comments_total 70
 thread_average_reaction_time_between_questions_tier_1 70
 thread_average_reaction_time_between_questions_tier_x 71
 thread_average_reaction_time_between_questions_total 71
 thread_average_response_to_comment_time_iama_host_tier_1 71
 thread_average_response_to_comment_time_iama_host_tier_x 71
 thread_average_response_to_comment_time_iama_host_total 71
 thread_average_response_to_question_time_iama_host_tier_1 71
 thread_average_response_to_question_time_iama_host_tier_x 71
 thread_average_response_to_question_time_iama_host_total 71
 thread_creation_time_stamp 71
 thread_downs 71
 thread_id 71
 thread_information 72
 thread_life_span_until_last_comment 72
 thread_life_span_until_last_question 72
 thread_num_comments_answered_by_iama_host_tier_1 72
 thread_num_comments_answered_by_iama_host_tier_x 72
 thread_num_comments_answered_by_iama_host_total 72
 thread_num_comments_tier_1 72
 thread_num_comments_tier_x 72
 thread_num_comments_total 72
 thread_num_comments_total_skewed 72
 thread_num_questions_answered_by_iama_host_tier_1 72
 thread_num_questions_answered_by_iama_host_tier_x 73
 thread_num_questions_answered_by_iama_host_total 73
 thread_num_questions_tier_1 73
 thread_num_questions_tier_x 73
 thread_num_questions_total 73
 thread_ups 73
 thread_year 73
 ar_analyzer.py 128
 argument_amount_of_top_quotes
 a_question_Answered_Yes_No_Extrema 45
 argument_calculation
 a_thread_Lifespan_N_Average_Commenttime 61
 a_thread_Lifespan_N_Average_Questiontime 66
 argument_crawl_type
 c_crawl_Threads_N_Comments 88
 argument_db_to_choose
 a_author_Information 28
 argument_hours_to_shift
 c_crawl_Threads_N_Comments 88
 argument_inverse_crawling
 c_crawl_Author_Information 76
 c_crawl_Differences 80
 argument_limit_crawling_amount
 c_crawl_Random_Author_Information 84
 argument_plot_time_unit
 a_iAMA_Commenttime_arr 33
 a_iAMA_Commenttime_med 39
 a_thread_Lifespan_N_Average_Commenttime 61
 a_thread_Lifespan_N_Average_Questiontime 66
 argument_sorting
 a_question_Answered_Yes_No_Extrema 45
 argument_tier_in_scope
 a_iAMA_Commenttime_arr 33

a_iAMA_Commenttime_med 39
a_question_Answered_Yes_No_Tier_Percentage 51
argument_year_beginning
a_iAMA_Commenttime_arr 33
a_iAMA_Commenttime_med 39
a_question_Answered_Yes_No_Extrema 45
a_question_Answered_Yes_No_Tier_Percentage 51
a_question_Tier_Distribution 56
a_thread_Lifespan_N_Average_Commenttime 61
a_thread_Lifespan_N_Average_Questiontime 66
c_crawl_Author_Information 76
c_crawl_Differences 80
c_crawl_Threads_N_Comments 89
d_create_Big_CSV_ar 94
d_create_Big_CSV_med 100
argument_year_end
c_crawl_Threads_N_Comments 89
argument_year_ending
a_iAMA_Commenttime_arr 33
a_iAMA_Commenttime_med 39
a_question_Answered_Yes_No_Extrema 45
a_question_Answered_Yes_No_Tier_Percentage 51
a_question_Tier_Distribution 56
a_thread_Lifespan_N_Average_Commenttime 61
a_thread_Lifespan_N_Average_Questiontime 66
c_crawl_Author_Information 76
c_crawl_Differences 80
d_create_Big_CSV_ar 94
d_create_Big_CSV_med 100
arr_of_values_f_authors
a__everything_Big_CSV_analyzer 8
author_amount_creation_iama_threads
a__everything_Big_CSV_analyzer 16
author_amount_creation_iama_threads_randomized
a__everything_Big_CSV_analyzer 16
author_amount_creation_other_threads
a__everything_Big_CSV_analyzer 16
author_amount_creation_other_threads_randomized
a__everything_Big_CSV_analyzer 16
author_amount_of_comments_except_iama
a__everything_Big_CSV_analyzer 16
author_amount_of_comments_except_iama_randomized
a__everything_Big_CSV_analyzer 16
author_amount_of_comments_iama
a__everything_Big_CSV_analyzer 16
author_amount_of_comments_iama_randomized
a__everything_Big_CSV_analyzer 17
author_author_birth_date
a__everything_Big_CSV_analyzer 17
author_author_birth_date_randomized
a__everything_Big_CSV_analyzer 17
author_author_comment_karma_amount
a__everything_Big_CSV_analyzer 17
author_author_comment_karma_amount_randomized
a__everything_Big_CSV_analyzer 17
author_author_link_karma_amount
a__everything_Big_CSV_analyzer 17
author_author_link_karma_amount_randomized
a__everything_Big_CSV_analyzer 17
author_author_name
a__everything_Big_CSV_analyzer 17
author_author_name_randomized
a__everything_Big_CSV_analyzer 17
author_comment_creation_every_x_sec
a__everything_Big_CSV_analyzer 17
author_comment_creation_every_x_sec_randomized
a__everything_Big_CSV_analyzer 18
author_information_iama
a__everything_Big_CSV_analyzer 18
author_information_iama_sampleset_randomized
a__everything_Big_CSV_analyzer 18
author_information_random
a__everything_Big_CSV_analyzer 18
author_information_random_sampleset_randomized
a__everything_Big_CSV_analyzer 18
author_thread_creation_every_x_sec
a__everything_Big_CSV_analyzer 18
author_thread_creation_every_x_sec_randomized
a__everything_Big_CSV_analyzer 18
author_time_acc_birth_first_iama_thread
a__everything_Big_CSV_analyzer 18
author_time_acc_birth_first_iama_thread_randomized
d
a__everything_Big_CSV_analyzer 18
author_time_diff_acc_creation_n_first_comment
a__everything_Big_CSV_analyzer 19
author_time_diff_acc_creation_n_first_comment_randomized
a__everything_Big_CSV_analyzer 19
author_time_diff_acc_creation_n_first_thread
a__everything_Big_CSV_analyzer 19
author_time_diff_acc_creation_n_first_thread_randomized
a__everything_Big_CSV_analyzer 19
average_means_of_values_f_authors
a__everything_Big_CSV_analyzer 8
average_means_of_values_f_threads
a__everything_Big_CSV_analyzer 8
ar_analyzer 69
bar_first_n_second_values_percentage
PlotlyBarChart::PlotlyBarChart 107
bar_percentages_values_1
PlotlyBarChart_5_Bars::PlotlyBarChart5Bars 111
bar_percentages_values_2
PlotlyBarChart_5_Bars::PlotlyBarChart5Bars 111
bar_percentages_values_3
PlotlyBarChart_5_Bars::PlotlyBarChart5Bars 112
bar_percentages_values_4
PlotlyBarChart_5_Bars::PlotlyBarChart5Bars 112
bar_percentages_values_5

PlotlyBarChart_5_Bars::PlotlyBarChart5Bars	112	mongo_DB_Threads_Instance	81
bar_value_description		start_crawling_for_diffs	80
PlotlyBarChart::PlotlyBarChart	107	c_crawl_Differences.py	131
PlotlyBarChart_5_Bars::PlotlyBarChart5Bars	112	c_crawl_Random_Author_Information	82
bar_x_axis_text		argument_limit_crawling_amount	84
PlotlyBarChart::PlotlyBarChart	107	calculate_time_difference	82
PlotlyBarChart_5_Bars::PlotlyBarChart5Bars	112	check_script_arguments	82
bar_x_axis_values		generate_data_now	82
PlotlyBarChart::PlotlyBarChart	107	get_author_information	83
PlotlyBarChart_5_Bars::PlotlyBarChart5Bars	112	initialize_mongo_db_parameters	84
bar_y_axis_fifth_values		mongo_db_client_instance	84
PlotlyBarChart_5_Bars::PlotlyBarChart5Bars	112	mongo_db_iama_author_collection	84
bar_y_axis_first_values		mongo_db_iama_author_collection_amount	84
PlotlyBarChart::PlotlyBarChart	107	mongo_db_iama_author_instance	84
PlotlyBarChart_5_Bars::PlotlyBarChart5Bars	112	mongo_db_random_author_collection	84
bar_y_axis_fourth_values		mongo_db_random_author_instance	85
PlotlyBarChart_5_Bars::PlotlyBarChart5Bars	112	reddit_instance	85
bar_y_axis_second_values		start_data_generation_for_analysis	84
PlotlyBarChart::PlotlyBarChart	107	c_crawl_Random_Author_Information.py	132
PlotlyBarChart_5_Bars::PlotlyBarChart5Bars	112	c_crawl_Threads_N_Comments	86
bar_y_axis_third_values		argument_crawl_type	88
PlotlyBarChart_5_Bars::PlotlyBarChart5Bars	112	argument_hours_to_shift	88
c_crawl_Author_Information	74	argument_year_beginning	89
argument_inverse_crawling	76	argument_year_end	89
argument_year_beginning	76	check_if_coll_in_db_already_exists_up2date	86
argument_year_ending	76	check_script_arguments	86
calculate_time_difference	74	convert_argument_year_to_epoch	87
check_script_arguments	74	crawl_comments	87
generate_data_now	74	crawl_data	87
get_author_information	75	crawl_threads	88
initialize_mongo_db_parameters	76	initialize_mongo_db_parameters	88
mongo_db_author_instance	76	mongo_DB_Client_Instance	89
mongo_db_client_instance	76	reddit_Instance	89
mongo_db_thread_collection	76	time_shift_difference	89
mongo_db_threads_instance	77	c_crawl_Threads_N_Comments.py	133
reddit_instance	77	calculate_ar_mean_answer_time_for_questions	
start_data_generation_for_analysis	76	a_iAMA_Commenttime_arr	30
year_actually_in_progress	77	a_iAMA_Commenttime_med	36
c_crawl_Author_Information.py	130	calculate_life_span	
c_crawl_Differences	78	d_create_Big_CSV_ar	90
argument_inverse_crawling	80	d_create_Big_CSV_med	96
argument_year_beginning	80	calculate_reaction_time_average	
argument_year_ending	80	d_create_Big_CSV_ar	91
check_if_collection_is_missing_in_comments_data		d_create_Big_CSV_med	97
base	78	calculate_t_tests_of_author_values	
check_if_collection_is_missing_in_threads_data		a_everything_Big_CSV_analyzer	8
base	78	calculate_time_difference	
check_script_arguments	79	a_iAMA_Commenttime_arr	30
crawl_missing_collection_into_comments_database		a_iAMA_Commenttime_med	36
base	79	a_question_Answered_Yes_No_Extrema	41
crawl_missing_collection_into_threads_database		a_thread_Lifespan_N_Average_Commenttime	59
base	79	a_thread_Lifespan_N_Average_Questiontime	63
initialize_mongo_db_parameters	80	c_crawl_Author_Information	74
mongo_DB_Client_Instance	80	c_crawl_Random_Author_Information	82
mongo_DB_Comments_Collection	80	d_create_Big_CSV_ar	91
mongo_DB_Comments_Instance	81	d_create_Big_CSV_med	97
mongo_DB_Thread_Collection	81	chart_title	

PlotlyBarChart::PlotlyBarChart	107	c_crawl_Random_Author_Information	82
PlotlyBarChart_5_Bars::PlotlyBarChart5Bars	113	c_crawl_Threads_N_Comments	86
check_if_coll_in_db_already_exists_up2date		d_create_Big_CSV_ar	93
c_crawl_Threads_N_Comments	86	d_create_Big_CSV_med	99
check_if_collection_is_missing_in_comments_database		color_1	
c_crawl_Differences	78	PlotlyBarChart_5_Bars::PlotlyBarChart5Bars	113
check_if_collection_is_missing_in_threads_database		color_1_border	
c_crawl_Differences	78	PlotlyBarChart_5_Bars::PlotlyBarChart5Bars	113
check_if_comment_has_been_answered_by_thread_author		color_2	
a_question_Answered_Yes_No_Extrema	42	PlotlyBarChart_5_Bars::PlotlyBarChart5Bars	113
d_create_Big_CSV_ar	91	color_2_border	
d_create_Big_CSV_med	97	PlotlyBarChart_5_Bars::PlotlyBarChart5Bars	113
check_if_comment_is_a_question		color_3	
a_iAMA_Commenttime_arr	30	PlotlyBarChart_5_Bars::PlotlyBarChart5Bars	113
a_iAMA_Commenttime_med	36	color_3_border	
a_question_Answered_Yes_No_Extrema	42	PlotlyBarChart_5_Bars::PlotlyBarChart5Bars	113
a_question_Answered_Yes_No_Tier_Percentage	48	color_4	
a_question_Tier_Distribution	53	PlotlyBarChart_5_Bars::PlotlyBarChart5Bars	113
d_create_Big_CSV_ar	92	color_4_border	
d_create_Big_CSV_med	98	PlotlyBarChart_5_Bars::PlotlyBarChart5Bars	113
check_if_comment_is_answer_from_thread_author		color_5	
a_iAMA_Commenttime_arr	31	PlotlyBarChart_5_Bars::PlotlyBarChart5Bars	113
a_iAMA_Commenttime_med	37	color_5_border	
a_question_Answered_Yes_No_Tier_Percentage	48	PlotlyBarChart_5_Bars::PlotlyBarChart5Bars	114
check_if_comment_is_not_from_thread_author		convert_argument_year_to_epoch	
a_iAMA_Commenttime_arr	31	c_crawl_Threads_N_Comments	87
a_iAMA_Commenttime_med	37	crawl_comments	
a_question_Answered_Yes_No_Tier_Percentage	48	c_crawl_Threads_N_Comments	87
check_if_comment_is_on_tier_1		crawl_data	
a_iAMA_Commenttime_arr	31	c_crawl_Threads_N_Comments	87
a_iAMA_Commenttime_med	37	crawl_missing_collection_into_comments_database	
a_question_Answered_Yes_No_Extrema	42	c_crawl_Differences	79
a_question_Answered_Yes_No_Tier_Percentage	48	crawl_missing_collection_into_threads_database	
a_question_Tier_Distribution	54	c_crawl_Differences	79
d_create_Big_CSV_ar	92	crawl_threads	
d_create_Big_CSV_med	98	c_crawl_Threads_N_Comments	88
check_if_comment_is_on_tier_1		create_question_list_containing_all_years	
a_iAMA_Commenttime_arr	31	a_question_Answered_Yes_No_Extrema	43
a_iAMA_Commenttime_med	37	current_year_question_list	
a_question_Answered_Yes_No_Tier_Percentage	49	a_question_Tier_Distribution	57
a_question_Tier_Distribution	54	d_create_Big_CSV_ar	90
d_create_Big_CSV_ar	92	add_actual_year_list_to_global_list	90
d_create_Big_CSV_med	98	argument_year_beginning	94
check_script_arguments		argument_year_ending	94
a_author_Information	27	calculate_life_span	90
a_iAMA_Commenttime_arr	31	calculate_reaction_time_average	91
a_iAMA_Commenttime_med	37	calculate_time_difference	91
a_question_Answered_Yes_No_Extrema	43	check_if_comment_has_been_answered_by_thread_author	91
a_question_Answered_Yes_No_Tier_Percentage	49	check_if_comment_is_a_question	92
a_question_Tier_Distribution	54	check_if_comment_is_not_from_thread_author	92
a_thread_Lifespan_N_Average_Commenttime	59	check_if_comment_is_on_tier_1	92
a_thread_Lifespan_N_Average_Questiontime	64	check_script_arguments	93
c_crawl_Author_Information	74	generate_data	93
c_crawl_Differences	79	initialize_mongo_db_parameters	93
		list_current_year	94

```

list_global_year 95
process_specific_thread 93
start_data_generation_for_analysis 94
write_csv_data 94
year_actually_in_progress 95
d_create_Big_CSV_ar.py 134
d_create_Big_CSV_med 96
    add_actual_year_list_to_global_list 96
    argument_year_beginning 100
    argument_year_ending 100
    calculate_life_span 96
    calculate_reaction_time_average 97
    calculate_time_difference 97
    check_if_comment_has_been_answered_by_thread
        _author 97
    check_if_comment_is_a_question 98
    check_if_comment_is_not_from_thread_author
        98
    check_if_comment_is_on_tier_1 98
    check_script_arguments 99
    generate_data 99
    initialize_mongo_db_parameters 99
    list_current_year 100
    list_global_year 101
    process_specific_thread 99
    start_data_generation_for_analysis 100
    write_csv_data 100
    year_actually_in_progress 101
d_create_Big_CSV_med.py 135
data_to_give_plotly
    a_iAMA_Commenttime_arr 34
    a_iAMA_Commenttime_med 40
    a_question_Answered_Yes_No_Extrema 45
    a_question_Answered_Yes_No_Tier_Percentage
        51
    a_question_Tier_Distribution 57
    a_thread_Lifespan_N_Average_Commenttime 61
    a_thread_Lifespan_N_Average_Questiontime 66
fill_bar_annotations
    PlotlyBarChart_5_Bars::PlotlyBarChart5Bars 109
fill_bar_description
    PlotlyBarChart::PlotlyBarChart 105
    PlotlyBarChart_5_Bars::PlotlyBarChart5Bars 109
fill_bar_percentages_values
    PlotlyBarChart::PlotlyBarChart 105
    PlotlyBarChart_5_Bars::PlotlyBarChart5Bars 109
fill_chart_title_description
    PlotlyBarChart::PlotlyBarChart 105
    PlotlyBarChart_5_Bars::PlotlyBarChart5Bars 110
fill_x_axis_list
    PlotlyBarChart::PlotlyBarChart 105
    PlotlyBarChart_5_Bars::PlotlyBarChart5Bars 110
fill_y_axis_answered_list
    PlotlyBarChart::PlotlyBarChart 106
fill_y_axis_unanswered_list
    PlotlyBarChart::PlotlyBarChart 106
fill_y_axis_values
    PlotlyBarChart_5_Bars::PlotlyBarChart5Bars 110
generate_chart
    PlotlyBarChart::PlotlyBarChart 106
    PlotlyBarChart_5_Bars::PlotlyBarChart5Bars 110
generate_data
    d_create_Big_CSV_ar 93
    d_create_Big_CSV_med 99
generate_data_now
    a_question_Answered_Yes_No_Extrema 43
    c_crawl_Author_Information 74
    c_crawl_Random_Author_Information 82
generate_data_to_be_analyzed
    a_iAMA_Commenttime_arr 32
    a_iAMA_Commenttime_med 38
    a_question_Answered_Yes_No_Tier_Percentage
        49
    a_question_Tier_Distribution 54
    a_thread_Lifespan_N_Average_Commenttime 59
    a_thread_Lifespan_N_Average_Questiontime 64
get_author_information
    c_crawl_Author_Information 75
    c_crawl_Random_Author_Information 83
global_question_list
    a_question_Answered_Yes_No_Tier_Percentage
        51
global_thread_list
    a_iAMA_Commenttime_arr 34
    a_iAMA_Commenttime_med 40
    a_thread_Lifespan_N_Average_Commenttime 62
    a_thread_Lifespan_N_Average_Questiontime 67
global_year_question_list
    a_question_Tier_Distribution 57
initialize_mongo_db_parameters
    a_author_Information 27
    a_iAMA_Commenttime_arr 32
    a_iAMA_Commenttime_med 38
    a_question_Answered_Yes_No_Extrema 43
    a_question_Answered_Yes_No_Tier_Percentage
        49
    a_question_Tier_Distribution 55
    a_thread_Lifespan_N_Average_Commenttime 59
    a_thread_Lifespan_N_Average_Questiontime 64
    c_crawl_Author_Information 76
    c_crawl_Differences 80
    c_crawl_Random_Author_Information 84
    c_crawl_Threads_N_Comments 88
    d_create_Big_CSV_ar 93
    d_create_Big_CSV_med 99
inplace
    a__everything_Big_CSV_analyzer 19
list_current_year
    d_create_Big_CSV_ar 94
    d_create_Big_CSV_med 100
list_global_year
    d_create_Big_CSV_ar 95
    d_create_Big_CSV_med 101
list_To_Be_Plotted

```

- a_iAMA_Commenttime_arr 34
- a_iAMA_Commenttime_med 40
- list_with_currents_year_infos
 - a_thread_Lifespan_N_Average_Commenttime 62
 - a_thread_Lifespan_N_Average_Questiontime 67
- main_method
 - PlotlyBarChart::PlotlyBarChart 106
 - PlotlyBarChart_5_Bars::PlotlyBarChart5Bars 110
- median_of_values_f_authors
 - a__everything_Big_CSV_analyzer 9
- median_of_values_f_authors_randomized
 - a__everything_Big_CSV_analyzer 9
- mongo_db_author_collection
 - a_author_Information 28
- mongo_db_author_collection_original
 - a_author_Information 28
- mongo_db_author_instance
 - a_author_Information 28
 - c_crawl_Author_Information 76
- mongo_db_client_instance
 - a_author_Information 28
 - c_crawl_Author_Information 76
 - c_crawl_Random_Author_Information 84
- mongo_DB_Client_Instance
 - a_iAMA_Commenttime_arr 34
 - a_iAMA_Commenttime_med 40
 - a_question_Answered_Yes_No_Extrema 46
 - a_question_Answered_Yes_No_Tier_Percentage 51
 - a_question_Tier_Distribution 57
 - a_thread_Lifespan_N_Average_Commenttime 62
 - a_thread_Lifespan_N_Average_Questiontime 67
 - c_crawl_Differences 80
 - c_crawl_Threads_N_Comments 89
- mongo_DB_Comments_Collection
 - c_crawl_Differences 80
- mongo_DB_Comments_Instance
 - a_iAMA_Commenttime_arr 34
 - a_iAMA_Commenttime_med 40
 - a_question_Answered_Yes_No_Extrema 46
 - a_question_Answered_Yes_No_Tier_Percentage 51
 - a_question_Tier_Distribution 57
 - a_thread_Lifespan_N_Average_Commenttime 62
 - a_thread_Lifespan_N_Average_Questiontime 67
 - c_crawl_Differences 81
- mongo_db_iama_author_collection
 - c_crawl_Random_Author_Information 84
- mongo_db_iama_author_collection_amount
 - c_crawl_Random_Author_Information 84
- mongo_db_iama_author_instance
 - c_crawl_Random_Author_Information 84
- mongo_db_random_author_collection
 - c_crawl_Random_Author_Information 84
- mongo_db_random_author_instance
 - c_crawl_Random_Author_Information 85
- mongo_db_thread_collection

- c_crawl_Author_Information 76
- mongo_DB_Thread_Collection
 - a_iAMA_Commenttime_arr 34
 - a_iAMA_Commenttime_med 40
 - a_question_Answered_Yes_No_Extrema 46
 - a_question_Answered_Yes_No_Tier_Percentage 52
 - a_question_Tier_Distribution 57
 - a_thread_Lifespan_N_Average_Commenttime 62
 - a_thread_Lifespan_N_Average_Questiontime 67
 - c_crawl_Differences 81
- mongo_db_threads_instance
 - c_crawl_Author_Information 77
- mongo_DB_Threads_Instance
 - a_iAMA_Commenttime_arr 34
 - a_iAMA_Commenttime_med 40
 - a_question_Answered_Yes_No_Extrema 46
 - a_question_Answered_Yes_No_Tier_Percentage 52
 - a_question_Tier_Distribution 57
 - a_thread_Lifespan_N_Average_Commenttime 62
 - a_thread_Lifespan_N_Average_Questiontime 67
 - c_crawl_Differences 81
- plot_generated_data
 - a_iAMA_Commenttime_arr 32
 - a_iAMA_Commenttime_med 38
 - a_question_Answered_Yes_No_Extrema 44
 - a_question_Answered_Yes_No_Tier_Percentage 50
 - a_question_Tier_Distribution 55
 - a_thread_Lifespan_N_Average_Commenttime 60
 - a_thread_Lifespan_N_Average_Questiontime 65
- PlotlyBarChart 102
- PlotlyBarChart.PlotlyBarChart 104
- PlotlyBarChart.py 136
- PlotlyBarChart::PlotlyBarChart
 - __init__ 104
 - bar_first_n_second_values_percentage 107
 - bar_value_description 107
 - bar_x_axis_text 107
 - bar_x_axis_values 107
 - bar_y_axis_first_values 107
 - bar_y_axis_second_values 107
 - chart_title 107
 - fill_bar_description 105
 - fill_bar_percentages_values 105
 - fill_chart_title_description 105
 - fill_x_axis_list 105
 - fill_y_axis_answered_list 106
 - fill_y_axis_unanswered_list 106
 - generate_chart 106
 - main_method 106
 - time_now_date 107
 - time_now_time 107
- PlotlyBarChart_5_Bars 103
- PlotlyBarChart_5_Bars.PlotlyBarChart5Bars 108
- PlotlyBarChart_5_Bars.py 137

PlotlyBarChart_5_Bars::PlotlyBarChart5Bars

__init__ 109
annotations_1 111
annotations_2 111
annotations_3 111
annotations_4 111
annotations_5 111
annotations_all 111
bar_percentages_values_1 111
bar_percentages_values_2 111
bar_percentages_values_3 112
bar_percentages_values_4 112
bar_percentages_values_5 112
bar_value_description 112
bar_x_axis_text 112
bar_x_axis_values 112
bar_y_axis_fifth_values 112
bar_y_axis_first_values 112
bar_y_axis_fourth_values 112
bar_y_axis_second_values 112
bar_y_axis_third_values 112
chart_title 113
color_1 113
color_1_border 113
color_2 113
color_2_border 113
color_3 113
color_3_border 113
color_4 113
color_4_border 113
color_5 113
color_5_border 114
fill_bar_annotations 109
fill_bar_description 109
fill_bar_percentages_values 109
fill_chart_title_description 110
fill_x_axis_list 110
fill_y_axis_values 110
generate_chart 110
main_method 110
time_now_date 114
time_now_time 114
prepare_data_for_comment_time
 a_thread_Lifespan_N_Average_Commenttime 60
 a_thread_Lifespan_N_Average_Questiontime 65
prepare_data_for_graph
 a_iAMA_Commenttime_arr 32
 a_iAMA_Commenttime_med 38
 a_question_Answered_Yes_No_Tier_Percentage 50
 a_question_Tier_Distribution 55
prepare_data_for_graph_life_span
 a_thread_Lifespan_N_Average_Commenttime 60
 a_thread_Lifespan_N_Average_Questiontime 65
prepare_dict_by_time_separation_for_comment_time
 a_thread_Lifespan_N_Average_Commenttime 60
 a_thread_Lifespan_N_Average_Questiontime 65

process_answered_questions_within_thread
 a_question_Answered_Yes_No_Extrema 44
process_specific_thread
 d_create_Big_CSV_ar 93
 d_create_Big_CSV_med 99
question_answered_by_iAMA_host
 a__everything_Big_CSV_analyzer 19
question_answering_distribution_tier1_tierx_tierany
 a_question_Answered_Yes_No_Tier_Percentage 50
question_distribution_tier1_tierx
 a_question_Tier_Distribution 55
question_information
 a__everything_Big_CSV_analyzer 19
question_information_list
 a_question_Answered_Yes_No_Extrema 46
question_overall_correlation
 a__everything_Big_CSV_analyzer 9
question_ups
 a__everything_Big_CSV_analyzer 19
random_author_amount_creation_iama_threads
 a__everything_Big_CSV_analyzer 19
random_author_amount_creation_iama_threads_randomized
 a__everything_Big_CSV_analyzer 19
random_author_amount_creation_other_threads
 a__everything_Big_CSV_analyzer 20
random_author_amount_creation_other_threads_randomized
 a__everything_Big_CSV_analyzer 20
random_author_amount_of_comments_except_iama
 a__everything_Big_CSV_analyzer 20
random_author_amount_of_comments_except_iama_randomized
 a__everything_Big_CSV_analyzer 20
random_author_amount_of_comments_iama
 a__everything_Big_CSV_analyzer 20
random_author_amount_of_comments_iama_randomized
 a__everything_Big_CSV_analyzer 20
random_author_author_birth_date
 a__everything_Big_CSV_analyzer 20
random_author_author_birth_date_randomized
 a__everything_Big_CSV_analyzer 20
random_author_author_comment_karma_amount
 a__everything_Big_CSV_analyzer 20
random_author_author_comment_karma_amount_randomized
 a__everything_Big_CSV_analyzer 21
random_author_author_link_karma_amount
 a__everything_Big_CSV_analyzer 21
random_author_author_link_karma_amount_randomized
 a__everything_Big_CSV_analyzer 21
random_author_author_name
 a__everything_Big_CSV_analyzer 21
random_author_author_name_randomized

a__everything_Big_CSV_analyzer 21
 random_author_comment_creation_every_x_sec
 a__everything_Big_CSV_analyzer 21
 random_author_comment_creation_every_x_sec_randomized
 a__everything_Big_CSV_analyzer 21
 random_author_thread_creation_every_x_sec
 a__everything_Big_CSV_analyzer 21
 random_author_thread_creation_every_x_sec_randomized
 a__everything_Big_CSV_analyzer 21
 random_author_time_acc_birth_first_iama_thread
 a__everything_Big_CSV_analyzer 22
 random_author_time_acc_birth_first_iama_thread_randomized
 a__everything_Big_CSV_analyzer 22
 random_author_time_diff_acc_creation_n_first_comment
 a__everything_Big_CSV_analyzer 22
 random_author_time_diff_acc_creation_n_first_comment_randomized
 a__everything_Big_CSV_analyzer 22
 random_author_time_diff_acc_creation_n_first_thread
 a__everything_Big_CSV_analyzer 22
 random_author_time_diff_acc_creation_n_first_thread_randomized
 a__everything_Big_CSV_analyzer 22
 relation_thread_amount_of_commentators_total_and_num_comments_answered_by_iama_host
 a__everything_Big_CSV_analyzer 9
 reddit_instance
 c_crawl_Author_Information 77
 c_crawl_Random_Author_Information 85
 reddit_Instance
 c_crawl_Threads_N_Comments 89
 relation_question_upvotes_with_amount_of_questions_answered_by_iama_host
 a__everything_Big_CSV_analyzer 9
 relation_thread_amount_of_questioners_total_and_num_questions_answered_by_iama_host
 a__everything_Big_CSV_analyzer 9
 relation_thread_amount_of_questions_and_amount_questions_answered_by_iama_host
 a__everything_Big_CSV_analyzer 10
 relation_thread_downvotes_and_iama_host_response_time_comments
 a__everything_Big_CSV_analyzer 10
 relation_thread_downvotes_and_iama_host_response_time_questions
 a__everything_Big_CSV_analyzer 10
 relation_thread_downvotes_with_amount_of_comments
 a__everything_Big_CSV_analyzer 10
 relation_thread_downvotes_with_amount_of_questions
 a__everything_Big_CSV_analyzer 11

relation_thread_lifespan_to_last_comment_and_amount_of_comments
 a__everything_Big_CSV_analyzer 11
 relation_thread_lifespan_to_last_comment_and_amount_of_questions
 a__everything_Big_CSV_analyzer 11
 relation_thread_lifespan_to_last_comment_and_iama_host_response_time_to_comments
 a__everything_Big_CSV_analyzer 11
 relation_thread_lifespan_to_last_comment_and_iama_host_response_time_to_questions
 a__everything_Big_CSV_analyzer 11
 relation_thread_lifespan_to_last_question_and_amount_of_comments
 a__everything_Big_CSV_analyzer 12
 relation_thread_lifespan_to_last_question_and_amount_of_question
 a__everything_Big_CSV_analyzer 12
 relation_thread_lifespan_to_last_question_and_iama_host_response_time_to_comments
 a__everything_Big_CSV_analyzer 12
 relation_thread_lifespan_to_last_question_and_iama_host_response_time_to_questions
 a__everything_Big_CSV_analyzer 12
 relation_thread_reaction_time_comments_and_amount_of_comments_the_iama_host_answered_to
 a__everything_Big_CSV_analyzer 13
 relation_thread_reaction_time_comments_and_amount_of_questions_the_iama_host_answered_to
 a__everything_Big_CSV_analyzer 13
 relation_thread_reaction_time_comments_and_iama_host_response_time_to_comments
 a__everything_Big_CSV_analyzer 13
 relation_thread_reaction_time_comments_and_iama_host_response_time_to_questions
 a__everything_Big_CSV_analyzer 13
 relation_thread_reaction_time_questions_and_amount_of_comments_the_iama_host_answered_to
 a__everything_Big_CSV_analyzer 14
 relation_thread_reaction_time_questions_and_amount_of_questions_the_iama_host_answered_to
 a__everything_Big_CSV_analyzer 14
 relation_thread_reaction_time_questions_and_iama_host_response_time_to_comments
 a__everything_Big_CSV_analyzer 14
 relation_thread_reaction_time_questions_and_iama_host_response_time_to_questions
 a__everything_Big_CSV_analyzer 14
 relation_thread_upvotes_and_iama_host_response_time_comments
 a__everything_Big_CSV_analyzer 15
 relation_thread_upvotes_and_iama_host_response_time_questions
 a__everything_Big_CSV_analyzer 15
 relation_thread_upvotes_with_amount_of_comments
 a__everything_Big_CSV_analyzer 15
 relation_thread_upvotes_with_amount_of_questions

a__everything_Big_CSV_analyzer 15
 sort_questions
 a_question_Answered_Yes_No_Extrema 44
 start_crawling_for_diffs
 c_crawl_Differences 80
 start_data_generation_for_analysis
 a_iAMA_Commenttime_arr 33
 a_iAMA_Commenttime_med 39
 a_question_Answered_Yes_No_Extrema 44
 a_question_Answered_Yes_No_Tier_Percentage
 50
 a_question_Tier_Distribution 56
 a_thread_Lifespan_N_Average_Commenttime 61
 a_thread_Lifespan_N_Average_Questiontime 66
 c_crawl_Author_Information 76
 c_crawl_Random_Author_Information 84
 d_create_Big_CSV_ar 94
 d_create_Big_CSV_med 100
 std_derivation
 ar_analyzer 69
 temp_time_difference_list
 a_thread_Lifespan_N_Average_Commenttime 62
 a_thread_Lifespan_N_Average_Questiontime 67
 thread_amount_of_commentators_tier_1
 a__everything_Big_CSV_analyzer 22
 ar_analyzer 69
 thread_amount_of_commentators_tier_x
 a__everything_Big_CSV_analyzer 22
 ar_analyzer 69
 thread_amount_of_commentators_total
 a__everything_Big_CSV_analyzer 22
 ar_analyzer 69
 thread_amount_of_questioners_tier_1
 a__everything_Big_CSV_analyzer 22
 ar_analyzer 69
 thread_amount_of_questioners_tier_x
 a__everything_Big_CSV_analyzer 23
 ar_analyzer 69
 thread_amount_of_questioners_total
 a__everything_Big_CSV_analyzer 23
 ar_analyzer 69
 thread_author
 a__everything_Big_CSV_analyzer 23
 ar_analyzer 70
 thread_average_comment_vote_score_tier_1
 a__everything_Big_CSV_analyzer 23
 ar_analyzer 70
 thread_average_comment_vote_score_tier_x
 a__everything_Big_CSV_analyzer 23
 ar_analyzer 70
 thread_average_comment_vote_score_total
 a__everything_Big_CSV_analyzer 23
 ar_analyzer 70
 thread_average_question_vote_score_tier_1
 a__everything_Big_CSV_analyzer 23
 ar_analyzer 70
 thread_average_question_vote_score_tier_x

a__everything_Big_CSV_analyzer 23
 ar_analyzer 70
 thread_average_question_vote_score_total
 a__everything_Big_CSV_analyzer 23
 ar_analyzer 70
 thread_average_reaction_time_between_comments_tier_1
 a__everything_Big_CSV_analyzer 23
 ar_analyzer 70
 thread_average_reaction_time_between_comments_tier_x
 a__everything_Big_CSV_analyzer 23
 ar_analyzer 70
 thread_average_reaction_time_between_comments_total
 a__everything_Big_CSV_analyzer 24
 ar_analyzer 70
 thread_average_reaction_time_between_questions_tier_1
 a__everything_Big_CSV_analyzer 24
 ar_analyzer 70
 thread_average_reaction_time_between_questions_tier_x
 a__everything_Big_CSV_analyzer 24
 ar_analyzer 71
 thread_average_reaction_time_between_questions_total
 a__everything_Big_CSV_analyzer 24
 ar_analyzer 71
 thread_average_response_to_comment_time_iama_host_tier_1
 a__everything_Big_CSV_analyzer 24
 ar_analyzer 71
 thread_average_response_to_comment_time_iama_host_tier_x
 a__everything_Big_CSV_analyzer 24
 ar_analyzer 71
 thread_average_response_to_comment_time_iama_host_total
 a__everything_Big_CSV_analyzer 24
 ar_analyzer 71
 thread_average_response_to_question_time_iama_host_tier_1
 a__everything_Big_CSV_analyzer 24
 ar_analyzer 71
 thread_average_response_to_question_time_iama_host_tier_x
 a__everything_Big_CSV_analyzer 24
 ar_analyzer 71
 thread_average_response_to_question_time_iama_host_total
 a__everything_Big_CSV_analyzer 24
 ar_analyzer 71
 thread_creation_time_stamp
 a__everything_Big_CSV_analyzer 24
 ar_analyzer 71
 thread_downs

a__everything_Big_CSV_analyzer 25
 ar_analyzer 71
 thread_id
 a__everything_Big_CSV_analyzer 25
 ar_analyzer 71
 thread_information
 a__everything_Big_CSV_analyzer 25
 ar_analyzer 72
 thread_life_span_until_last_comment
 a__everything_Big_CSV_analyzer 25
 ar_analyzer 72
 thread_life_span_until_last_question
 a__everything_Big_CSV_analyzer 25
 ar_analyzer 72
 thread_num_comments_answered_by_iamahost_tier_1
 a__everything_Big_CSV_analyzer 25
 ar_analyzer 72
 thread_num_comments_answered_by_iamahost_tier_x
 a__everything_Big_CSV_analyzer 25
 ar_analyzer 72
 thread_num_comments_answered_by_iamahost_total
 a__everything_Big_CSV_analyzer 25
 ar_analyzer 72
 thread_num_comments_tier_1
 a__everything_Big_CSV_analyzer 25
 ar_analyzer 72
 thread_num_comments_tier_x
 a__everything_Big_CSV_analyzer 25
 ar_analyzer 72
 thread_num_comments_total
 a__everything_Big_CSV_analyzer 26
 ar_analyzer 72
 thread_num_comments_total_skewed
 a__everything_Big_CSV_analyzer 26
 ar_analyzer 72
 thread_num_questions_answered_by_iamahost_tier_1
 a__everything_Big_CSV_analyzer 26
 ar_analyzer 72
 thread_num_questions_answered_by_iamahost_tier_x
 a__everything_Big_CSV_analyzer 26
 ar_analyzer 73
 thread_num_questions_answered_by_iamahost_total
 a__everything_Big_CSV_analyzer 26
 ar_analyzer 73
 thread_num_questions_tier_1
 a__everything_Big_CSV_analyzer 26

ar_analyzer 73
 thread_num_questions_tier_x
 a__everything_Big_CSV_analyzer 26
 ar_analyzer 73
 thread_num_questions_total
 a__everything_Big_CSV_analyzer 26
 ar_analyzer 73
 thread_overall_correlation
 a__everything_Big_CSV_analyzer 16
 thread_ups
 a__everything_Big_CSV_analyzer 26
 ar_analyzer 73
 thread_year
 a__everything_Big_CSV_analyzer 26
 ar_analyzer 73
 time_now_date
 PlotlyBarChart::PlotlyBarChart 107
 PlotlyBarChart_5_Bars::PlotlyBarChart5Bars 114
 time_now_time
 PlotlyBarChart::PlotlyBarChart 107
 PlotlyBarChart_5_Bars::PlotlyBarChart5Bars 114
 time_shift_difference
 c_crawl_Threads_N_Comments 89
 write_csv
 a_question_Answered_Yes_No_Tier_Percentage 51
 a_question_Tier_Distribution 56
 a_thread_Lifespan_N_Average_Commenttime 61
 a_thread_Lifespan_N_Average_Questiontime 66
 write_csv_and_count_unanswered
 a_question_Answered_Yes_No_Extrema 45
 write_csv_data
 a_author_Information 27
 a_iAMA_Commenttime_arr 33
 a_iAMA_Commenttime_med 39
 d_create_Big_CSV_ar 94
 d_create_Big_CSV_med 100
 year_actually_in_progress
 a_iAMA_Commenttime_arr 34
 a_iAMA_Commenttime_med 40
 a_question_Answered_Yes_No_Extrema 46
 a_question_Answered_Yes_No_Tier_Percentage 52
 a_question_Tier_Distribution 57
 a_thread_Lifespan_N_Average_Commenttime 62
 a_thread_Lifespan_N_Average_Questiontime 67
 c_crawl_Author_Information 77
 d_create_Big_CSV_ar 95
 d_create_Big_CSV_med 101
 year_question_list
 a_question_Answered_Yes_No_Tier_Percentage 52