

MakeUHappy Chat-bot

Создано системой Doxygen 1.8.20

1 MakeUNhappyChatBot	1
2 Алфавитный указатель пространств имен	3
2.1 Пространства имен	3
3 Алфавитный указатель классов	5
3.1 Классы	5
4 Список файлов	7
4.1 Файлы	7
5 Пространства имен	9
5.1 Пространство имен <code>bad_language</code>	9
5.2 Пространство имен <code>dialogpt</code>	9
5.3 Пространство имен <code>embeddings</code>	9
5.4 Пространство имен <code>prod</code>	9
5.4.1 Переменные	10
5.4.1.1 <code>args</code>	10
5.4.1.2 <code>help</code>	10
5.4.1.3 <code>parser</code>	10
5.4.1.4 <code>required</code>	10
5.4.1.5 <code>True</code>	10
5.5 Пространство имен <code>prod.worker</code>	11
5.6 Пространство имен <code>text</code>	11
5.6.1 Подробное описание	11
5.6.2 Функции	11
5.6.2.1 <code>extract_text()</code>	11
5.6.2.2 <code>get_text()</code>	12
5.7 Пространство имен <code>translate_emoji</code>	12
5.7.1 Функции	12
5.7.1.1 <code>de_emojify()</code>	12
5.7.1.2 <code>emoji_to_text()</code>	12
5.7.1.3 <code>eng_to_rus()</code>	13
5.7.2 Переменные	13
5.7.2.1 <code>translator</code>	13
5.8 Пространство имен <code>web</code>	13
5.8.1 Подробное описание	13
5.8.2 Функции	13
5.8.2.1 <code>get_updates()</code>	13
5.8.2.2 <code>send_message()</code>	14
5.8.3 Переменные	14
5.8.3.1 <code>api_url</code>	14
6 Классы	15
6.1 Класс <code>dialogpt.DialoGPT</code>	15

6.1.1	Подробное описание	15
6.1.2	Конструктор(ы)	15
6.1.2.1	<code>__init__()</code>	16
6.1.3	Методы	16
6.1.3.1	<code>generate()</code>	16
6.1.3.2	<code>get_length_param()</code>	16
6.1.3.3	<code>get_response()</code>	16
6.1.3.4	<code>process_text()</code>	17
6.1.3.5	<code>restart()</code>	17
6.1.4	Данные класса	17
6.1.4.1	<code>chat_history</code>	17
6.1.4.2	<code>model</code>	17
6.1.4.3	<code>tokenizer</code>	17
6.1.4.4	<code>user_input_size</code>	18
6.1.4.5	<code>window_size</code>	18
6.2	Класс <code>embeddings.EmbeddingCreator</code>	18
6.2.1	Подробное описание	18
6.2.2	Конструктор(ы)	18
6.2.2.1	<code>__init__()</code>	19
6.2.3	Методы	19
6.2.3.1	<code>get_by_index()</code>	19
6.2.3.2	<code>get_phrase_embedding()</code>	19
6.2.3.3	<code>make_embeddings()</code>	19
6.2.3.4	<code>run()</code>	20
6.2.3.5	<code>tokenize()</code>	20
6.2.3.6	<code>write_matrix()</code>	20
6.2.4	Данные класса	20
6.2.4.1	<code>device</code>	20
6.2.4.2	<code>embeddings</code>	20
6.2.4.3	<code>model</code>	21
6.2.4.4	<code>tokenized</code>	21
6.2.4.5	<code>tokenizer</code>	21
6.2.4.6	<code>words</code>	21
6.3	Класс <code>bad_language.Merger</code>	21
6.3.1	Подробное описание	22
6.3.2	Конструктор(ы)	22
6.3.2.1	<code>__init__()</code>	22
6.3.3	Методы	22
6.3.3.1	<code>__call__()</code>	22
6.3.4	Данные класса	22
6.3.4.1	<code>classifiers</code>	22
6.3.4.2	<code>is_soft</code>	22
6.4	Класс <code>embeddings.QuestionFinder</code>	23

6.4.1	Подробное описание	23
6.4.2	Конструктор(ы)	23
6.4.2.1	<code>__init__()</code>	23
6.4.3	Методы	23
6.4.3.1	<code>find_closest()</code>	23
6.4.4	Данные класса	24
6.4.4.1	<code>dim</code>	24
6.4.4.2	<code>embeddings</code>	24
6.4.4.3	<code>labels</code>	24
6.4.4.4	<code>num</code>	24
6.4.4.5	<code>p</code>	24
6.5	Класс <code>bad_language.SwearDetector</code>	25
6.5.1	Подробное описание	25
6.5.2	Конструктор(ы)	25
6.5.2.1	<code>__init__()</code>	25
6.5.3	Методы	25
6.5.3.1	<code>__call__()</code>	25
6.5.3.2	<code>clear()</code>	26
6.5.3.3	<code>find_swear()</code>	26
6.5.3.4	<code>has_swear()</code>	26
6.5.4	Данные класса	26
6.5.4.1	<code>blocklist</code>	26
6.5.4.2	<code>lemmatizer</code>	26
6.6	Класс <code>bad_language.ToxicClassifier</code>	27
6.6.1	Подробное описание	27
6.6.2	Конструктор(ы)	27
6.6.2.1	<code>__init__()</code>	27
6.6.3	Методы	27
6.6.3.1	<code>__call__()</code>	27
6.6.3.2	<code>how_toxic()</code>	28
6.6.3.3	<code>is_toxic()</code>	28
6.6.4	Данные класса	28
6.6.4.1	<code>toxic_bert</code>	28
6.6.4.2	<code>toxic_tokenizer</code>	28
6.7	Класс <code>prod.worker.Worker</code>	28
6.7.1	Подробное описание	29
6.7.2	Конструктор(ы)	29
6.7.2.1	<code>__init__()</code>	29
6.7.3	Методы	29
6.7.3.1	<code>work()</code>	29
6.7.3.2	<code>work_once()</code>	29
6.7.4	Данные класса	29
6.7.4.1	<code>chat_bots</code>	30

6.7.4.2 offset	30
6.7.4.3 token	30
7 Файлы	31
7.1 Файл data/swear.txt	31
7.2 Файл prod/__init__.py	31
7.3 Файл prod/utils/bad_language.py	31
7.4 Файл prod/utils/embeddings.py	32
7.5 Файл prod/utils/telegram/dialogpt.py	32
7.6 Файл prod/utils/telegram/text.py	32
7.7 Файл prod/utils/telegram/translate_emoji.py	32
7.8 Файл prod/utils/telegram/web.py	33
7.9 Файл prod/worker.py	33
7.10 Файл README.md	33
7.11 Файл requirements.txt	33
Предметный указатель	35

Глава 1

MakeUHappyChatBot

HSE Course Project: dialogue chat-bot

Глава 2

Алфавитный указатель пространств имен

2.1 Пространства имен

Полный список пространств имен.

bad_language	9
dialogpt	9
embeddings	9
prod	9
prod.worker	11
text	
Данное пространство имен содержит функции для обработки текстов, полученных по Telegram API	11
translate_emoji	12
web	
Содержит функции для взаимодействия с веб-сервисами	13

Глава 3

Алфавитный указатель классов

3.1 Классы

Классы с их кратким описанием.

dialogpt.DialogPT	15
embeddings.EmbeddingCreator	18
bad_language.Merger	21
embeddings.QuestionFinder	23
bad_language.SwearDetector	25
bad_language.ToxicClassifier	27
prod.worker.Worker	28

Глава 4

Список файлов

4.1 Файлы

Полный список файлов.

prod/	__init__.py	31
prod/	worker.py	33
prod/	utils/ bad_language.py	31
prod/	utils/ embeddings.py	32
prod/	utils/telegram/ dialogpt.py	32
prod/	utils/telegram/ text.py	32
prod/	utils/telegram/ translate_emoji.py	32
prod/	utils/telegram/ web.py	33

Глава 5

Пространства имен

5.1 Пространство имен `bad_language`

Классы

- class [Merger](#)
- class [SwearDetector](#)
- class [ToxicClassifier](#)

5.2 Пространство имен `dialogpt`

Классы

- class [DialoGPT](#)

5.3 Пространство имен `embeddings`

Классы

- class [EmbeddingCreator](#)
- class [QuestionFinder](#)

5.4 Пространство имен `prod`

Пространства имен

- [worker](#)

Переменные

- `parser = argparse.ArgumentParser()`
- `required`
- `True`
- `help`
- `args = parser.parse_args()`

5.4.1 Переменные

5.4.1.1 args

```
prod.args = parser.parse_args()
```

См. определение в файле `__init__.py` строка 7

5.4.1.2 help

```
prod.help
```

См. определение в файле `__init__.py` строка 6

5.4.1.3 parser

```
prod.parser = argparse.ArgumentParser()
```

См. определение в файле `__init__.py` строка 5

5.4.1.4 required

```
prod.required
```

См. определение в файле `__init__.py` строка 6

5.4.1.5 True

```
prod.True
```

См. определение в файле `__init__.py` строка 6

5.5 Пространство имен prod.worker

Классы

- class [Worker](#)

5.6 Пространство имен text

Данное пространство имен содержит функции для обработки текстов, полученных по Telegram API.

Функции

- def [extract_text](#) (message)
Извлекает текст из сообщения
- def [get_text](#) (message)
Получает и обрабатывает текст из сообщения

5.6.1 Подробное описание

Данное пространство имен содержит функции для обработки текстов, полученных по Telegram API.

5.6.2 Функции

5.6.2.1 [extract_text\(\)](#)

```
def text.extract_text (  
    message )
```

Извлекает текст из сообщения

Аргументы

message	Получаемое сообщение из Telegram API
---------	--------------------------------------

Возвращает

Текст из сообщения

См. определение в файле text.py строка 9

5.6.2.2 get_text()

```
def text.get_text (
    message )
```

Получает и обрабатывает текст из сообщения

Аргументы

message	Получаемое сообщение из Telegram API
---------	--------------------------------------

Возвращает

Обработанный текст

См. определение в файле text.py строка 23

5.7 Пространство имен translate_emoji

Функции

- def `eng_to_rus` (text)
- def `emoji_to_text` (emoji)
- def `de_emojify` (text)

Переменные

- `translator` = google_translator()

5.7.1 Функции

5.7.1.1 de_emojify()

```
def translate_emoji.de_emojify (
    text )
```

См. определение в файле translate_emoji.py строка 18

5.7.1.2 emoji_to_text()

```
def translate_emoji.emoji_to_text (
    emoji )
```

См. определение в файле translate_emoji.py строка 11

5.7.1.3 eng_to_rus()

```
def translate_emoji.eng_to_rus (
    text )
```

См. определение в файле translate_emoji.py строка 7

5.7.2 Переменные

5.7.2.1 translator

```
translate_emoji.translator = google_translator()
```

См. определение в файле translate_emoji.py строка 6

5.8 Пространство имен web

Содержит функции для взаимодействия с веб-сервисами

Функции

- def [get_updates](#) (token, offset)
Получает новые сообщения с сервера.
- def [send_message](#) (token, chat_id, text)
Отправляет сообщение в чат.

Переменные

- string [api_url](#) = 'https://api.telegram.org/'

5.8.1 Подробное описание

Содержит функции для взаимодействия с веб-сервисами

5.8.2 Функции

5.8.2.1 get_updates()

```
def web.get_updates (
    token,
    offset )
```

Получает новые сообщения с сервера.

Если сообщений нет, то возвращает пустой массив.

См. также

<https://core.telegram.org/bots/api#getupdates>

Аргументы

token	Токен для чат-бота
offset	Параметр offset в Telegram API

Возвращает

Массив с последними сообщениями

См. определение в файле web.py строка 15

5.8.2.2 send_message()

```
def web.send_message (
    token,
    chat_id,
    text )
```

Отправляет сообщение в чат.

См. также

<https://core.telegram.org/bots/api#sendmessage>

Аргументы

token	Токен для чат-бота
chat_id	id чата
text	Отправляемое сообщение

См. определение в файле web.py строка 25

5.8.3 Переменные

5.8.3.1 api_url

```
string web.api_url = 'https://api.telegram.org/'
```

См. определение в файле web.py строка 6

Глава 6

Классы

6.1 Класс dialogpt.DialogPT

Открытые члены

- `def __init__` (self, `window_size`=10)
- `str get_length_param` (self, `str text`)
- `def generate` (self, `bot_input_ids`, `num_return_sequences`=1, `max_length`=512, `no_repeat_ngram_size`=3, `do_sample`=True, `top_k`=50, `top_p`=0.9, `temperature`=0.6, `mask_token_id`=tokenizer.mask_token_id, `eos_token_id`=tokenizer.eos_token_id, `unk_token_id`=tokenizer.unk_token_id, `pad_token_id`=tokenizer.pad_token_id, `device`='cuda')
- `def process_text` (self, `input_user`)
- `def get_response` (self, `text`)
- `def restart` (self)

Открытые атрибуты

- `chat_history`
- `user_input_size`
- `window_size`

Статические открытые данные

- `tokenizer` = `AutoTokenizer.from_pretrained`("Grossmend/rudialogpt3_medium_based_on_gpt2")
- `model` = `AutoModelForCausalLM.from_pretrained`("Grossmend/rudialogpt3_medium_based_on_gpt2")

6.1.1 Подробное описание

См. определение в файле `dialogpt.py` строка 3

6.1.2 Конструктор(ы)

6.1.2.1 `__init__()`

```
def dialogpt.DialogPT.__init__ (
    self,
    window_size = 10 )
```

См. определение в файле dialogpt.py строка 10

6.1.3 Методы

6.1.3.1 `generate()`

```
def dialogpt.DialogPT.generate (
    self,
    bot_input_ids,
    num_return_sequences = 1,
    max_length = 512,
    no_repeat_ngram_size = 3,
    do_sample = True,
    top_k = 50,
    top_p = 0.9,
    temperature = 0.6,
    mask_token_id = tokenizer.mask_token_id,
    eos_token_id = tokenizer.eos_token_id,
    unk_token_id = tokenizer.unk_token_id,
    pad_token_id = tokenizer.pad_token_id,
    device = 'cuda' )
```

См. определение в файле dialogpt.py строка 27

6.1.3.2 `get_length_param()`

```
str dialogpt.DialogPT.get_length_param (
    self,
    str text )
```

См. определение в файле dialogpt.py строка 15

6.1.3.3 `get_response()`

```
def dialogpt.DialogPT.get_response (
    self,
    text )
```

См. определение в файле dialogpt.py строка 91

6.1.3.4 process_text()

```
def dialogpt.DialoGPT.process_text (
    self,
    input_user )
```

См. определение в файле dialogpt.py строка 61

6.1.3.5 restart()

```
def dialogpt.DialoGPT.restart (
    self )
```

См. определение в файле dialogpt.py строка 121

6.1.4 Данные класса

6.1.4.1 chat_history

```
dialogpt.DialoGPT.chat_history
```

См. определение в файле dialogpt.py строка 11

6.1.4.2 model

```
dialogpt.DialoGPT.model = AutoModelForCausalLM.from_pretrained("Grossmend/rudialogpt3_medium_based_on_gpt2") [static]
```

См. определение в файле dialogpt.py строка 5

6.1.4.3 tokenizer

```
dialogpt.DialoGPT.tokenizer = AutoTokenizer.from_pretrained("Grossmend/rudialogpt3_medium_based_on_gpt2") [static]
```

См. определение в файле dialogpt.py строка 4

6.1.4.4 user_input_size

`dialogpt.DialogPT.user_input_size`

См. определение в файле `dialogpt.py` строка 12

6.1.4.5 window_size

`dialogpt.DialogPT.window_size`

См. определение в файле `dialogpt.py` строка 13

Объявления и описания членов класса находятся в файле:

- `prod/utils/telegram/dialogpt.py`

6.2 Класс `embeddings.EmbeddingCreator`

Открытые члены

- `def __init__ (self, model_name_or_path="sberbank-ai/rugpt3small_based_on_gpt2")`
- `def tokenize (self, data)`
- `def make_embeddings (self, tokenized_data)`
- `def write_matrix (self, data, filename='embeddings')`
- `def run (self, path_or_word_list, output_filename='embeddings')`
- `def get_phrase_embedding (self, phrase)`
- `def get_by_index (self, index)`

Открытые атрибуты

- `words`
- `device`
- `tokenizer`
- `model`
- `tokenized`
- `embeddings`

6.2.1 Подробное описание

См. определение в файле `embeddings.py` строка 7

6.2.2 Конструктор(ы)

6.2.2.1 __init__()

```
def embeddings.EmbeddingCreator.__init__(  
    self,  
    model_name_or_path = "sberbank-ai/rugpt3small_based_on_gpt2" )
```

For usage examples go to ~/junk/run_this_all.ipynb

См. определение в файле embeddings.py строка 8

6.2.3 Методы

6.2.3.1 get_by_index()

```
def embeddings.EmbeddingCreator.get_by_index (  
    self,  
    index )
```

См. определение в файле embeddings.py строка 63

6.2.3.2 get_phrase_embedding()

```
def embeddings.EmbeddingCreator.get_phrase_embedding (  
    self,  
    phrase )
```

См. определение в файле embeddings.py строка 57

6.2.3.3 make_embeddings()

```
def embeddings.EmbeddingCreator.make_embeddings (  
    self,  
    tokenized_data )
```

См. определение в файле embeddings.py строка 21

6.2.3.4 run()

```
def embeddings.EmbeddingCreator.run (
    self,
    path_or_word_list,
    output_filename = 'embeddings' )
```

См. определение в файле embeddings.py строка 33

6.2.3.5 tokenize()

```
def embeddings.EmbeddingCreator.tokenize (
    self,
    data )
```

См. определение в файле embeddings.py строка 17

6.2.3.6 write_matrix()

```
def embeddings.EmbeddingCreator.write_matrix (
    self,
    data,
    filename = 'embeddings' )
```

См. определение в файле embeddings.py строка 29

6.2.4 Данные класса

6.2.4.1 device

```
embeddings.EmbeddingCreator.device
```

См. определение в файле embeddings.py строка 13

6.2.4.2 embeddings

```
embeddings.EmbeddingCreator.embeddings
```

См. определение в файле embeddings.py строка 49

6.2.4.3 `model`

`embeddings.EmbeddingCreator.model`

См. определение в файле `embeddings.py` строка 15

6.2.4.4 `tokenized`

`embeddings.EmbeddingCreator.tokenized`

См. определение в файле `embeddings.py` строка 45

6.2.4.5 `tokenizer`

`embeddings.EmbeddingCreator.tokenizer`

См. определение в файле `embeddings.py` строка 14

6.2.4.6 `words`

`embeddings.EmbeddingCreator.words`

См. определение в файле `embeddings.py` строка 11

Объявления и описания членов класса находятся в файле:

- `prod/utils/embeddings.py`

6.3 Класс `bad_language.Merger`

Открытые члены

- `def __init__(self, classifiers=[ToxicClassifier\(\), SwearDetector\(\)], is_soft=False)`
- `def __call__(self, sentence)`

Открытые атрибуты

- `classifiers`
- `is_soft`

6.3.1 Подробное описание

См. определение в файле `bad_language.py` строка 63

6.3.2 Конструктор(ы)

6.3.2.1 `__init__()`

```
def bad_language.Merger.__init__(
    self,
    classifiers = [ToxicClassifier(), SwearDetector()],
    is_soft = False )
```

См. определение в файле `bad_language.py` строка 64

6.3.3 Методы

6.3.3.1 `__call__()`

```
def bad_language.Merger.__call__(
    self,
    sentence )
```

См. определение в файле `bad_language.py` строка 68

6.3.4 Данные класса

6.3.4.1 `classifiers`

```
bad_language.Merger.classifiers
```

См. определение в файле `bad_language.py` строка 65

6.3.4.2 `is_soft`

```
bad_language.Merger.is_soft
```

См. определение в файле `bad_language.py` строка 66

Объявления и описания членов класса находятся в файле:

- [prod/utls/bad_language.py](#)

6.4 Класс `embeddings.QuestionFinder`

Открытые члены

- `def __init__` (self, embeddings_path_or_matrix)
- `def find_closest` (self, embedding, num_neighbours=1, remoteness=0)

Открытые атрибуты

- `embeddings`
- `num`
- `dim`
- `labels`
- `p`

6.4.1 Подробное описание

См. определение в файле `embeddings.py` строка 74

6.4.2 Конструктор(ы)

6.4.2.1 `__init__()`

```
def embeddings.QuestionFinder.__init__ (
    self,
    embeddings_path_or_matrix )
```

For usage examples go to `~/junk/run_this_all.ipynb`

См. определение в файле `embeddings.py` строка 75

6.4.3 Методы

6.4.3.1 `find_closest()`

```
def embeddings.QuestionFinder.find_closest (
    self,
    embedding,
    num_neighbours = 1,
    remoteness = 0 )
```

См. определение в файле `embeddings.py` строка 104

6.4.4 Данные класса

6.4.4.1 dim

`embeddings.QuestionFinder.dim`

См. определение в файле `embeddings.py` строка 87

6.4.4.2 embeddings

`embeddings.QuestionFinder.embeddings`

См. определение в файле `embeddings.py` строка 77

6.4.4.3 labels

`embeddings.QuestionFinder.labels`

См. определение в файле `embeddings.py` строка 89

6.4.4.4 num

`embeddings.QuestionFinder.num`

См. определение в файле `embeddings.py` строка 86

6.4.4.5 p

`embeddings.QuestionFinder.p`

См. определение в файле `embeddings.py` строка 91

Объявления и описания членов класса находятся в файле:

- `prod/utils/embeddings.py`

6.5 Класс bad_language.SwearDetector

Открытые члены

- def `__init__` (self, path_or_list='../data/swear.txt', use_stemming=True)
- def `__call__` (self, sentence)
- def `clear` (self, sentence)
- def `has_swear` (self, sentence)
- def `find_swear` (self, sentence)

Открытые атрибуты

- `lemmatizer`
- `blocklist`

6.5.1 Подробное описание

См. определение в файле bad_language.py строка 23

6.5.2 Конструктор(ы)

6.5.2.1 `__init__`()

```
def bad_language.SwearDetector.__init__ (
    self,
    path_or_list = '../data/swear.txt ',
    use_stemming = True )
```

См. определение в файле bad_language.py строка 24

6.5.3 Методы

6.5.3.1 `__call__`()

```
def bad_language.SwearDetector.__call__ (
    self,
    sentence )
```

См. определение в файле bad_language.py строка 37

6.5.3.2 clear()

```
def bad_language.SwearDetector.clear (
    self,
    sentence )
```

См. определение в файле `bad_language.py` строка 40

6.5.3.3 find_swear()

```
def bad_language.SwearDetector.find_swear (
    self,
    sentence )
```

См. определение в файле `bad_language.py` строка 52

6.5.3.4 has_swear()

```
def bad_language.SwearDetector.has_swear (
    self,
    sentence )
```

См. определение в файле `bad_language.py` строка 43

6.5.4 Данные класса

6.5.4.1 blocklist

```
bad_language.SwearDetector.blocklist
```

См. определение в файле `bad_language.py` строка 29

6.5.4.2 lemmatizer

```
bad_language.SwearDetector.lemmatizer
```

См. определение в файле `bad_language.py` строка 25

Объявления и описания членов класса находятся в файле:

- [prod/utils/bad_language.py](#)

6.6 Класс `bad_language.ToxicClassifier`

Открытые члены

- `def __init__` (self, model_path='sismetanin/rubert-toxic-pikabu-2ch')
- `def __call__` (self, message)
- `def is_toxic` (self, message)
- `def how_toxic` (self, message)

Открытые атрибуты

- `toxic_tokenizer`
- `toxic_bert`

6.6.1 Подробное описание

См. определение в файле `bad_language.py` строка 7

6.6.2 Конструктор(ы)

6.6.2.1 `__init__`()

```
def bad_language.ToxicClassifier.__init__ (
    self,
    model_path = 'sismetanin/rubert-toxic-pikabu-2ch ' )
```

См. определение в файле `bad_language.py` строка 8

6.6.3 Методы

6.6.3.1 `__call__`()

```
def bad_language.ToxicClassifier.__call__ (
    self,
    message )
```

См. определение в файле `bad_language.py` строка 12

6.6.3.2 how_toxic()

```
def bad_language.ToxicClassifier.how_toxic (
    self,
    message )
```

См. определение в файле bad_language.py строка 18

6.6.3.3 is_toxic()

```
def bad_language.ToxicClassifier.is_toxic (
    self,
    message )
```

См. определение в файле bad_language.py строка 15

6.6.4 Данные класса

6.6.4.1 toxic_bert

```
bad_language.ToxicClassifier.toxic_bert
```

См. определение в файле bad_language.py строка 10

6.6.4.2 toxic_tokenizer

```
bad_language.ToxicClassifier.toxic_tokenizer
```

См. определение в файле bad_language.py строка 9

Объявления и описания членов класса находятся в файле:

- [prod/utils/bad_language.py](#)

6.7 Класс prod.worker.Worker

Открытые члены

- `def __init__ (self, token)`
- `def work_once (self)`
- `def work (self)`

Открытые атрибуты

- `chat_bots`
- `offset`
- `token`

6.7.1 Подробное описание

См. определение в файле `worker.py` строка 6

6.7.2 Конструктор(ы)

6.7.2.1 `__init__()`

```
def prod.worker.Worker.__init__(  
    self,  
    token )
```

См. определение в файле `worker.py` строка 7

6.7.3 Методы

6.7.3.1 `work()`

```
def prod.worker.Worker.work(  
    self )
```

См. определение в файле `worker.py` строка 37

6.7.3.2 `work_once()`

```
def prod.worker.Worker.work_once(  
    self )
```

См. определение в файле `worker.py` строка 12

6.7.4 Данные класса

6.7.4.1 chat_bots

`prod.worker.Worker.chat_bots`

См. определение в файле `worker.py` строка 8

6.7.4.2 offset

`prod.worker.Worker.offset`

См. определение в файле `worker.py` строка 9

6.7.4.3 token

`prod.worker.Worker.token`

См. определение в файле `worker.py` строка 10

Объявления и описания членов класса находятся в файле:

- `prod/worker.py`

Глава 7

Файлы

7.1 Файл data/swear.txt

7.2 Файл prod/__init__.py

Пространства имен

- `prod`

Переменные

- `prod.parser = argparse.ArgumentParser()`
- `prod.required`
- `prod.True`
- `prod.help`
- `prod.args = parser.parse_args()`

7.3 Файл prod/utils/bad_language.py

Классы

- `class bad_language.ToxicClassifier`
- `class bad_language.SwearDetector`
- `class bad_language.Merger`

Пространства имен

- `bad_language`

7.4 Файл prod/utils/embeddings.py

Классы

- class `embeddings.EmbeddingCreator`
- class `embeddings.QuestionFinder`

Пространства имен

- `embeddings`

7.5 Файл prod/utils/telegram/dialogpt.py

Классы

- class `dialogpt.DialoGPT`

Пространства имен

- `dialogpt`

7.6 Файл prod/utils/telegram/text.py

Пространства имен

- `text`

Данное пространство имен содержит функции для обработки текстов, полученных по Telegram API.

Функции

- def `text.extract_text` (message)
Извлекает текст из сообщения
- def `text.get_text` (message)
Получает и обрабатывает текст из сообщения

7.7 Файл prod/utils/telegram/translate_emoji.py

Пространства имен

- `translate_emoji`

Функции

- def `translate_emoji.eng_to_rus` (text)
- def `translate_emoji.emoji_to_text` (emoji)
- def `translate_emoji.de_emojify` (text)

Переменные

- `translate_emoji.translator` = `google_translator()`

7.8 Файл prod/utils/telegram/web.py

Пространства имен

- `web`
Содержит функции для взаимодействия с веб-сервисами

Функции

- def `web.get_updates` (token, offset)
Получает новые сообщения с сервера.
- def `web.send_message` (token, chat_id, text)
Отправляет сообщение в чат.

Переменные

- string `web.api_url` = 'https://api.telegram.org/'

7.9 Файл prod/worker.py

Классы

- class `prod.worker.Worker`

Пространства имен

- `prod.worker`

7.10 Файл README.md

7.11 Файл requirements.txt

Предметный указатель

- `-- call --`
 - `bad_language.Merger`, [22](#)
 - `bad_language.SwearDetector`, [25](#)
 - `bad_language.ToxicClassifier`, [27](#)
 - `-- init --`
 - `bad_language.Merger`, [22](#)
 - `bad_language.SwearDetector`, [25](#)
 - `bad_language.ToxicClassifier`, [27](#)
 - `dialogpt.DialoGPT`, [15](#)
 - `embeddings.EmbeddingCreator`, [18](#)
 - `embeddings.QuestionFinder`, [23](#)
 - `prod.worker.Worker`, [29](#)
- `api_url`
 - `web`, [14](#)
- `args`
 - `prod`, [10](#)
- `bad_language`, [9](#)
- `bad_language.Merger`, [21](#)
 - `-- call --`, [22](#)
 - `-- init --`, [22](#)
 - `classifiers`, [22](#)
 - `is_soft`, [22](#)
- `bad_language.SwearDetector`, [25](#)
 - `-- call --`, [25](#)
 - `-- init --`, [25](#)
 - `blocklist`, [26](#)
 - `clear`, [25](#)
 - `find_swear`, [26](#)
 - `has_swear`, [26](#)
 - `lemmatizer`, [26](#)
- `bad_language.ToxicClassifier`, [27](#)
 - `-- call --`, [27](#)
 - `-- init --`, [27](#)
 - `how_toxic`, [27](#)
 - `is_toxic`, [28](#)
 - `toxic_bert`, [28](#)
 - `toxic_tokenizer`, [28](#)
- `blocklist`
 - `bad_language.SwearDetector`, [26](#)
- `chat_bots`
 - `prod.worker.Worker`, [29](#)
- `chat_history`
 - `dialogpt.DialoGPT`, [17](#)
- `classifiers`
 - `bad_language.Merger`, [22](#)
- `clear`
 - `bad_language.SwearDetector`, [25](#)
- `data/swear.txt`, [31](#)
- `de_emojify`
 - `translate_emoji`, [12](#)
- `device`
 - `embeddings.EmbeddingCreator`, [20](#)
- `dialogpt`, [9](#)
- `dialogpt.DialoGPT`, [15](#)
 - `-- init --`, [15](#)
 - `chat_history`, [17](#)
 - `generate`, [16](#)
 - `get_length_param`, [16](#)
 - `get_response`, [16](#)
 - `model`, [17](#)
 - `process_text`, [16](#)
 - `restart`, [17](#)
 - `tokenizer`, [17](#)
 - `user_input_size`, [17](#)
 - `window_size`, [18](#)
- `dim`
 - `embeddings.QuestionFinder`, [24](#)
- `embeddings`, [9](#)
 - `embeddings.EmbeddingCreator`, [20](#)
 - `embeddings.QuestionFinder`, [24](#)
- `embeddings.EmbeddingCreator`, [18](#)
 - `-- init --`, [18](#)
 - `device`, [20](#)
 - `embeddings`, [20](#)
 - `get_by_index`, [19](#)
 - `get_phrase_embedding`, [19](#)
 - `make_embeddings`, [19](#)
 - `model`, [20](#)
 - `run`, [19](#)
 - `tokenize`, [20](#)
 - `tokenized`, [21](#)
 - `tokenizer`, [21](#)
 - `words`, [21](#)
 - `write_matrix`, [20](#)
- `embeddings.QuestionFinder`, [23](#)
 - `-- init --`, [23](#)
 - `dim`, [24](#)
 - `embeddings`, [24](#)
 - `find_closest`, [23](#)
 - `labels`, [24](#)
 - `num`, [24](#)
 - `p`, [24](#)
- `emoji_to_text`
 - `translate_emoji`, [12](#)
- `eng_to_rus`

- translate_emoji, 12
- extract_text
 - text, 11
- find_closest
 - embeddings.QuestionFinder, 23
- find_swear
 - bad_language.SwearDetector, 26
- generate
 - dialogpt.DialoGPT, 16
- get_by_index
 - embeddings.EmbeddingCreator, 19
- get_length_param
 - dialogpt.DialoGPT, 16
- get_phrase_embedding
 - embeddings.EmbeddingCreator, 19
- get_response
 - dialogpt.DialoGPT, 16
- get_text
 - text, 11
- get_updates
 - web, 13
- has_swear
 - bad_language.SwearDetector, 26
- help
 - prod, 10
- how_toxic
 - bad_language.ToxicClassifier, 27
- is_soft
 - bad_language.Merger, 22
- is_toxic
 - bad_language.ToxicClassifier, 28
- labels
 - embeddings.QuestionFinder, 24
- lemmatizer
 - bad_language.SwearDetector, 26
- make_embeddings
 - embeddings.EmbeddingCreator, 19
- model
 - dialogpt.DialoGPT, 17
 - embeddings.EmbeddingCreator, 20
- num
 - embeddings.QuestionFinder, 24
- offset
 - prod.worker.Worker, 30
- p
 - embeddings.QuestionFinder, 24
- parser
 - prod, 10
- process_text
 - dialogpt.DialoGPT, 16
- prod, 9
 - args, 10
 - help, 10
 - parser, 10
 - required, 10
 - True, 10
 - prod.worker, 11
 - prod.worker.Worker, 28
 - __init__, 29
 - chat_bots, 29
 - offset, 30
 - token, 30
 - work, 29
 - work_once, 29
 - prod/__init__.py, 31
 - prod/utils/bad_language.py, 31
 - prod/utils/embeddings.py, 32
 - prod/utils/telegram/dialogpt.py, 32
 - prod/utils/telegram/text.py, 32
 - prod/utils/telegram/translate_emoji.py, 32
 - prod/utils/telegram/web.py, 33
 - prod/worker.py, 33
 - README.md, 33
 - required
 - prod, 10
 - requirements.txt, 33
 - restart
 - dialogpt.DialoGPT, 17
 - run
 - embeddings.EmbeddingCreator, 19
 - send_message
 - web, 14
 - text, 11
 - extract_text, 11
 - get_text, 11
 - token
 - prod.worker.Worker, 30
 - tokenize
 - embeddings.EmbeddingCreator, 20
 - tokenized
 - embeddings.EmbeddingCreator, 21
 - tokenizer
 - dialogpt.DialoGPT, 17
 - embeddings.EmbeddingCreator, 21
 - toxic_bert
 - bad_language.ToxicClassifier, 28
 - toxic_tokenizer
 - bad_language.ToxicClassifier, 28
 - translate_emoji, 12
 - de_emojiify, 12
 - emoji_to_text, 12
 - eng_to_rus, 12
 - translator, 13
 - translator
 - translate_emoji, 13
 - True
 - prod, 10

user_input_size
 dialogpt.DialoGPT, [17](#)

web, [13](#)
 api_url, [14](#)
 get_updates, [13](#)
 send_message, [14](#)

window_size
 dialogpt.DialoGPT, [18](#)

words
 embeddings.EmbeddingCreator, [21](#)

work
 prod.worker.Worker, [29](#)

work_once
 prod.worker.Worker, [29](#)

write_matrix
 embeddings.EmbeddingCreator, [20](#)