src

src package

Subpackages

src.browser package

Submodules

src.browser.browser module

src.browser.browser_options module

src.browser.google_selenium_browser module

src.browser.mixins module

src.browser.selenium_browser module

src.browser.selenium_utils module

Module contents

src.captcha package

Submodules

src.captcha.captcha_api module

src.captcha.captcha_service module

src.captcha.google_captcha module

Module contents

src.config package

Submodules

src.config.app module

class **src.config.app.ConfigApp**

Базовые классы: `object`

CAPTCHA_SOURCE= *'google-auth'*

CAPTCHA_TIMEOUT= *10*

START_URL_AUTH
= *'https://accounts.google.com/ServiceLogin?hl=ru&passive=true&continue=https://www.google.com/&*

GOOGLE_URL= *'https://accounts.google.com'*

START_URL_NAME
= *'https://accounts.google.com/signin/v2/usernamerecovery?theme=glif&flowName=GlifWebSignIn&flowEntry=S*

MAX_ONE_SCREEN_ENTRY= *4*

MAX_GET_ELEMENT_ATTEMPTS= *3*

MAX_LEAVE_MAIN_PAGE_ATTEMPTS= *3*

src.config.settings module

Module contents

src.exceptions package

Submodules

src.exceptions.google_exceptions module

Module contents

src.interfaces package

Submodules

src.interfaces.abstract_browser module

class **src.interfaces.abstract_browser.AbstractBrowser**(**args*, ***kwargs*)

Базовые классы: `object`

`proxy_extension`: `AbstractProxyExtension`

`is_started`: `bool`

`browser_prepared`: `bool`

`options`: `ChromeOptions`

`headless`: `bool`

`window_size`: `list` [`int`]

`implicitly_wait_delay`: `float`

`explicit_wait_delay`: `float`

abstract **`start_browser`**()

Тип результата:

`None`

abstract **`close_browser`**()

Завершает работу web драйвера.

Тип результата:

`None`

abstract **`get`**(*url*)

Переходит по переданному url и ждет загрузки страницы.

Параметры:

url (`str`) – url-адрес страницы.

Тип результата:

`None`

abstract property **`page_source`**: *str*

Возвращает текущую html-страницу.

Результат:

html код страницы в формате str.

abstract **get_element**(*by*, *selector*)

Тип результата:

`WebElement`

abstract **get_element_and_clear**(*by*, *selector*)

Тип результата:

`WebElement`

abstract **get_loaded_element**(*by*, *selector*)

Тип результата:

`WebElement`

abstract **get_element_and_click**(*by*, *selector*)

Находит элемент на странице и переходит по нему (click). Имеет задержку для ожидания обновления страницы.

Параметры:

- **by** (`By`) – локатор, определяет стратегию поиска.
- **selector** (`str`) – ключ поиска.

Тип результата:

`None`

get_element_and_set_data(*by*, *selector*, *data*)

Находит элемент на странице и вводит в него данные.

Параметры:

- **by** (`By`) – локатор, определяет стратегию поиска.
- **selector** (`str`) – ключ поиска.
- **data** (`str`) – данные для ввода.

Тип результата:

`None`

abstract **get_element_set_data_and_enter**(*by*, *selector*, *data*)

Находит элемент на странице, вводит данные и отправляет сигнал нажатия Enter. Имеет задержку для ожидания обновления страницы.

Параметры:

- **by** (`By`) – локатор, определяет стратегию поиска.
- **selector** (`str`) – ключ поиска.
- **data** (`str`) – данные для ввода.

Тип результата:

`None`

abstract **get_current_url**()

Возвращает текущий URL

Тип результата:
`str`

src.interfaces.abstract_captcha_service module

class **src.interfaces.abstract_captcha_service.AbstractCaptchaService**

Базовые классы: `ABC`

abstract **solve_captcha**(*image*, *timeout*)

Тип результата:
`dict` | `None`

abstract **solve_report**(*task_id*, *correct*)

Тип результата:
`dict` | `str` | `None`

src.interfaces.abstract_extension module

class **src.interfaces.abstract_extension.AbstractProxyExtension**

Базовые классы: `ABC`

abstract **prepare**(*host*, *port*, *user*, *password*)

Тип результата:
`None`

abstract property **directory**: *str*

src.interfaces.abstract_google_captcha_service module

class **src.interfaces.abstract_google_captcha_service.AbstractGoogleCaptchaService**

Базовые классы: `AbstractCaptchaService`

image_tag: `tuple` [`By` , `str`]

input_tag: `tuple` [`By` , `str`]

src.interfaces.abstract_response_adapter module

class src.interfaces.abstract_response_adapter.ResponseAdapter

Базовые классы: `ABC`

abstract
static **cast**(*response*)

Тип результата:

`list`[`dict`[`str`, `str` | `list`]]

src.interfaces.abstract_screen_dispatcher module

class src.interfaces.abstract_screen_dispatcher.AbstractScreenDispatcher

Базовые классы: `ABC`

Собирает и возвращает данные со страницы.

abstract **get_data**(*web_browser*)

Тип результата:

`dict`[`str`, `list` | `bool`] | `None`

src.interfaces.abstract_screens_repository module

src.interfaces.abstract_telegram_api module

class src.interfaces.abstract_telegram_api.AbstractTelegramAPI

Базовые классы: `ABC`

abstract **send_file**(*file*, *send_as=None*)

Отправляет файл в телеграм.

Параметры:

- **file** (`str`) – Путь к файлу.
- **send_as** (`str` | `None`) – Изменить имя отправляемого файла.

Тип результата:

`str`

Результат:

Ответ клиента.

abstract **send_message**(*message*)

Отправляет сообщение в телеграм.

Параметры:

message (`str`) – Сообщение.

Тип результата:

`str`

Результат:

Ответ клиента.

src.interfaces.abstract_telegram_bot module

class **src.interfaces.abstract_telegram_bot.AbstractTelegramBot**

Базовые классы: `ABC`

message_prefix = ''

abstract **send_files_from_path**(*path*, *message*)

Тип результата:

`None`

src.interfaces.event module

class **src.interfaces.event.Event**(*iterable=()*, */*)

Базовые классы: `list`

src.interfaces.utils module

class **src.interfaces.utils.SingletonABCMeta**(*name*, *bases*, *namespace*, */*, ***kwargs*)

Базовые классы: `ABCMeta`

Module contents

src.logger package

Submodules

src.logger.logger module

Module contents

src.logic package

Subpackages

src.logic.adapters package

Submodules

src.logic.adapters.response_auth module

class src.logic.adapters.response_auth.ResponseAuthAdapter

Базовые классы: `ResponseAdapter`

static cast(*response*)

Тип результата:
`list`[`dict`[`str`, `str` | `list`]]

static remove_duplicates(*iterable*)

Тип результата:
`list`

static phones_clean(*phones*)

Тип результата:
`list`[`str`]

src.logic.adapters.response_name module

class src.logic.adapters.response_name.ResponseNameAdapter

Базовые классы: `ResponseAdapter`

static cast(*response*)

Тип результата:
`list`[`dict`[`str`, `str` | `list`]]

Module contents

src.logic.google package

Submodules

src.logic.google.configured_screen module

src.logic.google.connections_auth module

src.logic.google.connections_name module

src.logic.google.screen module

src.logic.google.screen_dispatchers module

src.logic.google.screen_explorer module

src.logic.google.screen_explorer_auth module

src.logic.google.screen_explorer_name module

src.logic.google.screen_repository module

src.logic.google.search_manager module

src.logic.google.search_manager_auth module

src.logic.google.search_manager_name module

Module contents

src.logic.keydb package

Submodules

src.logic.keydb.fieldXML_auth module

src.logic.keydb.fieldXML_name module

Module contents

src.logic.thread package

Submodules

src.logic.thread.exception_handler module

Module contents

Submodules

src.logic.utils module

src.logic.utils.random_string(*length*)

Тип результата:
`str`

src.logic.utils.get_domain_url(*url*)

Тип результата:
`str`

Module contents

src.runners package

Submodules

src.runners.chosen_tasks module

src.runners.fill_name_queue module

src.runners.fill_queue module

src.runners.main module

src.runners.main_name module

src.runners.mobile_operators module

src.runners.mobile_operators.get_random_mobile_phone()

Тип результата:
`str`

src.runners.mobile_operators.get_random_mobile_phones(*count*)

Тип результата:
`list`[`str`]

Module contents

src.telegram package

Submodules

src.telegram.bot module

src.telegram.telegram_api module

Модуль для работы с телеграм ботом.

URL адрес для работы с телеграм ботом: https://api.telegram.org/bot<token>/METHOD_NAME

Отправить сообщение: https://api.telegram.org/bot<token>/sendMessage Отправить файл: https://api.telegram.org/bot<token>/sendDocument

В теле сообщения обязательно передаем «chat_id»: self.chat_id

Пример отпавки файла из терминала: curl -v -F «chat_id=569502265» -F document=@/Users/users/Desktop/file.txt https://api.telegram.org/bot<TOKEN>/sendDocument

class **src.telegram.telegram_api.TelegramAPI**(*token*, *chat_id*)

Базовые классы: `AbstractTelegramAPI`

send_file(*file*, *send_as=None*)

Отправляет файл в телеграм.

Параметры:

- **file** (`str`) – Путь к файлу.
- **send_as** (`str` | `None`) – Изменить имя отправляемого файла.

Тип результата:

`str`

Результат:

Ответ клиента.

send_message(*message*)

Отправляет сообщение в телеграм.

Параметры:

message (`str`) – Сообщение.

Тип результата:

`str`

Результат:

Ответ клиента.

Module contents

Module contents

start_auth module

start_name module