PykeBot2

Release 2.4.2

Jonathan Decker

May 01, 2022

CONTENTS:

CHAPTER

ONE

PYKEBOT2

1.1 PykeBot2 package

1.1.1 Subpackages

PykeBot2.backend package

Subpackages

PykeBot2.backend.stalker package

Submodules

PykeBot2.backend.stalker.battlefy module

Uses undocumented Battlefy API to scrape tournament participants.

author Jonathan Decker

async PykeBot2.backend.stalker.battlefy.**stalk_battlefy_tournament**(*battlefy_url: str*)

Uses undocumented Battlefy API to scrape tournament participants. :param battlefy_url: A valid url to a battlefy tournament. :type battlefy_url: str :return: a TeamList object containing all Teams and Players of the given tournament. :rtype: TeamList

PykeBot2.backend.stalker.op_gg_rank module

Uses op.gg for player rank stalking and offers further functions for adding ranks to teams and calculating average rankings.

author Jonathan Decker

async PykeBot2.backend.stalker.op_gg_rank.**add_player_rank**(*player: PykeBot2.models.data_models.Player*, *session: Optional[aiohttp.client.ClientSession] = None*)

Description Calls stalk player op gg using the summoner name of the given Player and adds a Rank obj to the Player.

Parameters

- **player** (Player) – A Player obj with a summoner name.
- **session** (*aiohttp.ClientSession*) – When a session already exits, it should be reused as much as possible for better performance.

Returns None

Return type None

async PykeBot2.backend.stalker.op_gg_rank.**add_team_list_list_ranks**(*team_list_list: PykeBot2.models.data_models.TeamListList*, *session: Optional[aiohttp.client.ClientSession] = None*)

Description Calls add team list ranks for each team list in the given team list list obj.

Parameters

- **team_list_list** (TeamListList) – A team list list with a list of team lists.
- **session** (*aiohttp.ClientSession*) – When a session already exits, it should be reused as much as possible for better performance.

Returns None

Return type None

async PykeBot2.backend.stalker.op_gg_rank.**add_team_list_ranks**(*team_list: PykeBot2.models.data_models.TeamList*, *session: Optional[aiohttp.client.ClientSession] = None*)

Description Calls add team ranks for each team in the given team list obj.

Parameters

- **team_list** (TeamList) – A team list with a list of teams.
- **session** (*aiohttp.ClientSession*) – When a session already exits, it should be reused as much as possible for better performance.

Returns None

Return type None

async PykeBot2.backend.stalker.op_gg_rank.**add_team_ranks**(*team: PykeBot2.models.data_models.Team*, *session: Optional[aiohttp.client.ClientSession] = None*)

Description Calls add player rank for each player of the given team. Also sets the average and max team rank.

Parameters

- **team** (Team) – A team with a list of players.
- **session** (*aiohttp.ClientSession*) – When a session already exits, it should be reused as much as possible for better performance.

Returns None

Return type None

PykeBot2.backend.stalker.op_gg_rank.**calc_average_and_max_team_rank**(*team: PykeBot2.models.data_models.Team*)

Description Calculates the average and maximum rank of a given team.

If a player is unranked or the rank is unknown, then the player is ignored for the average. :param team: A team obj with players who have Rank objs. :type team: Team :return: None :rtype: None

async PykeBot2.backend.stalker.op_gg_rank.**stalk_player_op_gg**(*sum_name: str*, *session: Optional[aiohttp.client.ClientSession] = None*)

Description Uses aiohttp to find the rank of a single player from op.gg.

Parameters

- **sum_name** (*str*) – Summoner name can be taken from Player object inside Team objects.
- **session** (*aiohttp.ClientSession*) – When a session already exits, it should be reused as much as possible for better performance.

Returns A string representation of the Rank, should be used to create a Rank obj.

Return type str

PykeBot2.backend.stalker.prime_league module

Handles scraping of the prime league page. :author: Jonathan Decker

PykeBot2.backend.stalker.prime_league.**filter_group_links**(*link*)

Helper function to filter out non group links. :param link: Str, a web link :return: Bool, true if the link contains a keyword and leads to a group, false if not

PykeBot2.backend.stalker.prime_league.**filter_team_links**(*link*)

Helper function to filter out non team links. :param link: Str, a web link :return: Bool, true if the link contains a teams, false if not

async PykeBot2.backend.stalker.prime_league.**stalk_prime_league_group**(*prime_league_group_link: str*, *session: Optional[aiohttp.client.ClientSession] = None*, *headless: bool = True*)

Description Uses aiohttp requests to stalk a prime league group.

Also contains an extra case for the swiss starter group. :param prime_league_group_link: A valid link to a prime league group. :type prime_league_group_link: str :param session: When a session already exits, it should be reused as much as possible for better performance. :type session: aiohttp.ClientSession :param headless: The swiss starter group regroup requires the selenium webriver. Use for debugging. :type headless: bool :return: TeamList object containing all gathered information. :rtype: TeamList

async PykeBot2.backend.stalker.prime_league.**stalk_prime_league_season**(*prime_league_season_link: str*, *headless=True*)

Description Uses Selenium to open the link and gather all group links from it,

further calls stalk prime league group on all groups. :param prime_league_season_link: A valid link to a prime league season. :type prime_league_season_link: str :param headless: Whether the browser should be headless or not. Use head for debugging purposes. :type headless: bool :return: TeamListList object containing all gathered information. :rtype: TeamListList

async PykeBot2.backend.stalker.prime_league.**stalk_prime_league_team**(*prime_league_team_link: str*, *session: Optional[aiohttp.client.ClientSession] = None*)

Description Uses aiohttp requests to stalk a prime league team.

Parameters

- **prime_league_team_link** –
- **session** (*aiohttp.ClientSession*) – When a session already exits, it should be reused as much as possible for better performance.

Returns Team object containing all the gathered information.

Return type *Team*

PykeBot2.backend.stalker.riot_api_rank module

Uses the riot api to fetch the ranks of players

author Jonathan Decker

class PykeBot2.backend.stalker.riot_api_rank.**RateLimiter**(*session: aiohttp.client.ClientSession*)

Bases: object

add_new_tokens()

async **get**(**args*, ***kwargs*)

max_tokens = 500

rate = 50

regen_after = 60.0

request_counter = 0

start_time = 0

async **wait_for_tokens**()

async PykeBot2.backend.stalker.riot_api_rank.**add_player_rank**(*player: PykeBot2.models.data_models.Player*, *api_token: str*, *session=None*)

Description Calls stalk player riot using the summoner name of the given Player and adds a Rank obj to the Player.

Parameters

- **player** (Player) – A Player obj with a summoner name.
- **api_token** (*str*) – Valid Riot api token.

- **session** (*aiohttp.ClientSession or RateLimiter*) – When a session already exits, it should be reused as much as possible for better performance.

Returns None

Return type None

async PykeBot2.backend.stalker.riot_api_rank.**add_team_list_list_ranks**(*team_list_list: PykeBot2.models.data_models.TeamListList*, *api_token: str*, *session=None*)

Description Calls add team list ranks for each team list in the given team list list obj.

Parameters

- **team_list_list** (*TeamListList*) – A team list list with a list of team lists.
- **api_token** (*str*) – Valid Riot api token.
- **session** (*aiohttp.ClientSession or RateLimiter*) – When a session already exits, it should be reused as much as possible for better performance.

Returns None

Return type None

async PykeBot2.backend.stalker.riot_api_rank.**add_team_list_ranks**(*team_list: PykeBot2.models.data_models.TeamList*, *api_token: str*, *session=None*)

Description Calls add team ranks for each team in the given team list obj.

Parameters

- **team_list** (*TeamList*) – A team list with a list of teams.
- **api_token** (*str*) – Valid Riot api token.
- **session** (*aiohttp.ClientSession or RateLimiter*) – When a session already exits, it should be reused as much as possible for better performance.

Returns None

Return type None

async PykeBot2.backend.stalker.riot_api_rank.**add_team_ranks**(*team: PykeBot2.models.data_models.Team*, *api_token: str*, *session=None*)

Description Calls add player rank for each player of the given team. Also sets the average and max team rank.

Parameters

- **team** (*Team*) – A team with a list of players.
- **api_token** (*str*) – Valid Riot api token.
- **session** (*aiohttp.ClientSession or RateLimiter*) – When a session already exits, it should be reused as much as possible for better performance.

Returns None

Return type None

PykeBot2.backend.stalker.riot_api_rank.**calc_average_and_max_team_rank**(*team: PykeBot2.models.data_models.Team*)

Description Calculates the average and maximum rank of a given team.

If a player is unranked or the rank is unknown, then the player is ignored for the average. :param team: A team obj with players who have Rank objs. :type team: Team :return: None :rtype: None

async PykeBot2.backend.stalker.riot_api_rank.**stalk_player_riot_api**(*sum_name: str*, *api_token: str*, *session=None*) → str

Uses the riot api to find the soloQ ranking of the given player. :param sum_name: Summoner name of the player. :type sum_name: str :param api_token: Valid Riot api token. :type api_token: str :param session: When a session already exits, it should be reused as much as possible for better performance. :type session: aiohttp.ClientSession or RateLimiter :return: String representation of the rank of the player. :rtype: str

PykeBot2.backend.stalker.summoners_inn module

Handles scraping of summoners inn with focus on the hausarrest cup. :author: Jonathan Decker

PykeBot2.backend.stalker.summoners_inn.**filter_team_links**(*link*)

Helper function to filter out non team links. :param link: Str, a web link :return: Bool, true if the link contains a teams, false if not

async PykeBot2.backend.stalker.summoners_inn.**stalk_summoners_inn_cup**(*summoners_inn_cup_link: str*, *session: Optional[aiohttp.client.ClientSession] = None*)

Takes a link to a summoners inn cup, extracts all team links and uses the prime league team stalker, to stalk the teams. :param summoners_inn_cup_link: A valid link to a summoners inn cup like Hausarrest. :type summoners_inn_cup_link: str :param session: A reusable ClientSession :type session: aiohttp.ClientSession :return: A team list containing all teams of the cup. :rtype: TeamList

PykeBot2.backend.stalker.toornament module

Handles scraping of the toornament page. :author: Jonathan Decker

async PykeBot2.backend.stalker.toornament.**stalk_toornament_team**(*toornament_team_link: str*, *session: Optional[aiohttp.client.ClientSession] = None*)

Stalks all players in the given team and returns a Team Object :raises ServerErrorResponseError, NotFoundResponseError :param toornament_team_link: Link to a toornament team page :type toornament_team_link: str :param session: A session that can be reused, if none is given, a new one will be created :type session: aiohttp.ClientSession :return: team containing all players of the given team :rtype: Team

async PykeBot2.backend.stalker.toornament.**stalk_toornament_tournament**(*toornament_link: str*)

Stalks all teams signed up for the given toornament and returns a TeamList Object :raises ServerErrorResponseError, NotFoundResponseError :param toornament_link: url to a tournament on toornament :type toornament_link: str :return: TeamList, containing the Team obj for each signed up team :rtype: TeamList

PykeBot2.backend.stalker.toornament_api module

Handles scraping of the toornament page using the official toornament API.

author Jonathan Decker

PykeBot2.backend.stalker.toornament_api.**fetch_groups**(*toornament_link: str*, *api_token: str*)

PykeBot2.backend.stalker.toornament_api.**fetch_matches**(*toornament_link: str*, *api_token: str*)

PykeBot2.backend.stalker.toornament_api.**fetch_participants**(*toornament_link: str*, *api_token: str*) → List[dict]

PykeBot2.backend.stalker.toornament_api.**fetch_stages**(*toornament_link: str*, *api_token: str*)

PykeBot2.backend.stalker.toornament_api.**fetch_tournament**(*toornament_link: str*) → str

PykeBot2.backend.stalker.toornament_api.**parse_participants**(*participants: List[dict]*, *tournament_name: str*) → *PykeBot2.models.data_models.TeamList*

Parses the given participant list and tournament name to create a TeamList object. :param participants: Expects a participant list created from the toornament api. :type participants: list[dict] :param tournament_name: The name of the tournament. :type tournament_name: str :return: A TeamList object created from the given participant list and tournament name. :rtype: TeamList

async PykeBot2.backend.stalker.toornament_api.**stalk_toornament_api_tournament**(*toornament_link: str*) → *PykeBot2.models.data_models.TeamL*

Stalks teams in the given toornament link using the toornament api. A valid token in a file called ToornamentToken must be in the working directory for this. :param toornament_link: A valid link to a toornament tournament :type toornament_link: str :return: A TeamList object containing all teams from the given tournament :rtype: TeamList

Module contents

Submodules

PykeBot2.backend.backend_master module

Main Backend Module, responsible for calling stalker functions for query data or calling the database interface. When finished sets the payload field of the query with data object.

author Jonathan Decker

async PykeBot2.backend.backend_master.**backend_loop**(*forward_queue: asyncio.queues.Queue*, *backend_queue: asyncio.queues.Queue*)

Description Main Coroutine for the backend. Responsible for calling stalker functions.

Parameters

- **forward_queue** (*asyncio.Queue*) – The Queue which is handled by the forwarder of the main event loop.
- **backend_queue** (*asyncio.Queue*) – The Queue that is handled by this Coroutine.

Returns None

Return type None

async PykeBot2.backend.backend_master.**call_rank_stalker**(*payload:* PykeBot2.models.data_models.Payload, *use_api=False*)

Description Checks the payload type and calls the correct rank stalker.

Parameters

- **payload** (Payload) – The Payload object returned from stalking. Submitting Error, Message or Player will cause an error.
- **use_api** (*Bool*) – If set to true, will try to use the Riot api to fetch ranks instead of op.gg.

Returns None

Return type None

PykeBot2.backend.backend_master.**create_error**(*query:* PykeBot2.models.query.Query, *content: str*)

Description Creates an error message from the content and adds it to the query,

further sets query forward to to frontend and next step to format. :param query: The handled query, which encountered an error. :type query: Query :param content: The error message to be displayed, should usually include str(query). :type content: str :return: None :rtype: None

PykeBot2.backend.backend_master.**determine_stalker**(*query:* PykeBot2.models.query.Query)

Description Checks the url for keywords to determine the correct stalker.

Parameters **query** (Query) – The handled Query.

Returns The stalker function fitting the url.

Return type function (coroutine)

PykeBot2.backend.backend_master.**logger = <Logger pb_logger (WARNING)>**

Helper dictionaries used to map identifier in the url to stalker functions.

Module contents

PykeBot2.frontend package

Submodules

PykeBot2.frontend.command_interpreter module

Responsible for understanding the raw command in a given Query and setting next step, flags and data accordingly.

author Jonathan Decker

PykeBot2.frontend.command_interpreter.**create_error**(*query:* PykeBot2.models.query.Query, *content: str*)

Description Creates an error message from the content and adds it to the query,

further sets query forward to to frontend and next step to format. :param query: The handled query, which encountered an error. :type query: Query :param content: The error message to be displayed, should usually include str(query). :type content: str :return: None :rtype: None

PykeBot2.frontend.command_interpreter.**interpret_command**(*query:* PykeBot2.models.query.Query)

Description Interprets the raw command of the Query and sets next step, flags and data accordingly.

Parameters **query** (Query) – The handled Query.

Returns None

Return type None

PykeBot2.frontend.console_interface module

PykeBot2.frontend.discord_interface module

author Jonathan Decker

class PykeBot2.frontend.discord_interface.**PykeBot**(**args*, ***kwargs*)

Bases: discord.ext.commands.bot.Bot

Description Overwrites the standard Discord Bot to add fields for the incoming and outgoing Queues as well as

background tasks. The prefix for co

forward_queue: asyncio.queues.Queue

output_queue: asyncio.queues.Queue

async output_queue_listener()

Description Coroutine that handles the discord output queue and sends messages based on the incoming queries.

The query objects is not further forwarded. :return: None :rtype: None

PykeBot2.frontend.discord_interface.**chunk_message**(*out_raw: str*)

PykeBot2.frontend.discord_interface.**initiate_query**(*message: discord.message.Message*)

Description Creates query object from the given message and submits it to the forward query.

Parameters **message** (*discord.Message*) – A valid message, usually captured by the on_message event.

Returns None

Return type None

async PykeBot2.frontend.discord_interface.**on_message**(*message: discord.message.Message*)

Description Event triggered when a message is send to a viewed channel.

If the message starts with the prefix (standard: '.pb'), a query is created from the message and forwarded. Another exception is {prefix} ping which results in an immediate Pong instead of a Query. :param message: The message object that triggered the event. :type message: discord.Message :return: None :rtype: None

async PykeBot2.frontend.discord_interface.**on_ready**()

Description Event triggered when the Discord Bot is ready. Logs the event and sets the presence for the bot.

Returns None

Return type None

async PykeBot2.frontend.discord_interface.**run_discord_bot_loop**(*forward_queue: asyncio.queues.Queue*, *output_queue: asyncio.queues.Queue*)

Description Main Coroutine for the Discord Bot interface.

Handles the final setup steps and starts the main Coroutine for Discord Bot. :param forward_queue: Where queries created from incoming messages are put. :type forward_queue: asyncio.Queue :param output_queue: The message from queries in this Queue will be send back via the interface. :type output_queue: asyncio.Queue :return: None :rtype: None

PykeBot2.frontend.frontend_master module

Main Frontend module, responsible for calling command interpreter on new queries, calling output formatter and checking the context before sending a query to the respective interface.

author Jonathan Decker

async PykeBot2.frontend.frontend_master.**frontend_loop**(*forward_queue: asyncio.queues.Queue*, *frontend_queue: asyncio.queues.Queue*)

Description Main Coroutine for the frontend. Responsible for calling command interpreter and output formatter.

Parameters

- **forward_queue** (*asyncio.Queue*) – The Queue which is handled by the forwarder of the main event loop.
- **frontend_queue** (*asyncio.Queue*) – The Queue that is handled by this Coroutine.

Returns None

Return type None

async PykeBot2.frontend.frontend_master.**loop_back_dummy**(*forward_queue: asyncio.queues.Queue*, *frontend_queue: asyncio.queues.Queue*)

Description A dummy Coroutine that sends incoming queries back to the discord interface.

Parameters

- **forward_queue** (*asyncio.Queue*) – The forward_queue which should be handled by the query_forwarder.
- **frontend_queue** (*asyncio.Queue*) – The Queue that is handled by this Coroutine.

Returns None

Return type None

PykeBot2.frontend.output_formatter module

Responsible for creating the output message from the payload by calling str messages, respecting flags.

author Jonathan Decker

PykeBot2.frontend.output_formatter.**create_error**(*query:* PykeBot2.models.query.Query, *content: str*)

Description Creates an error message from the content and adds it to the query,

further sets query forward to to frontend and next step to format. :param query: The handled query, which encountered an error. :type query: Query :param content: The error message to be displayed, should usually include str(query). :type content: str :return: None :rtype: None

PykeBot2.frontend.output_formatter.**format_error**(*query:* PykeBot2.models.query.Query)

Description Formats error payloads, and sets output message.

Parameters **query** (Query) – The handled Query.

Returns None

Return type None

PykeBot2.frontend.output_formatter.**format_message**(*query:* PykeBot2.models.query.Query)

Description Formats message payloads, and sets output message.

Parameters **query** (Query) – The handled Query.

Returns None

Return type None

PykeBot2.frontend.output_formatter.**format_payload**(*query:* PykeBot2.models.query.Query)

Description Takes the Payload from the Query and calls to str based on context and flags and saves it to output message.

Parameters **query** (Query) – The handled Query.

Returns None

Return type None

Module contents

PykeBot2.models package

Submodules

PykeBot2.models.data_models module

Provides data models for Player, Team, Team list and Team list list objects. Further also Error and Message objects are also defined. All of them are subclasses of Payload which is used in Query. Finally also Rank is defined.

All payload subclasses define to str and to discord str functions which are used by the output formatter Player, Team, Team list and Team list list also define extended to str functions which display further information on the Players and the Rank if possible.

author Jonathan Decker

class PykeBot2.models.data_models.**Error**(*content: str*)

Bases: *PykeBot2.models.data_models.Payload*

content: str

discord_str()

class PykeBot2.models.data_models.**Message**(*content: str*)

Bases: *PykeBot2.models.data_models.Payload*

content: str

discord_str()

class PykeBot2.models.data_models.**Payload**

Bases: object

discord_extended_str()

discord_str()

extended_str()

class PykeBot2.models.data_models.**Player**(*sum_name*)

Bases: *PykeBot2.models.data_models.Payload*

Saves information on a single league account

discord_str()

opgg: str

rank: *PykeBot2.models.data_models.Rank* = Rank()

summoner_name: str

class PykeBot2.models.data_models.**Rank**(*rank_string: Optional[str] = None*, *rank_int: Optional[int] = None*)

Bases: object

Saves a player rank as string and integer

rank_int = -1

rank_string = 'Unknown'

class PykeBot2.models.data_models.**Team**(*name*, *players*)

Bases: *PykeBot2.models.data_models.Payload*

Saves information for a team of league players

average_rank: *PykeBot2.models.data_models.Rank* = None

build_op_gg_multi_link()

construct a valid op.gg multi link for the given summoner names :return: String, url to the multilink for the given names

discord_extended_str()

discord_str()

extended_str()

max_rank: *PykeBot2.models.data_models.Rank* = None

multi_link: str

name: str

players: List[*PykeBot2.models.data_models.Player*]

top5_average_rank = None

class PykeBot2.models.data_models.**TeamList**(*name*, *teams*)

Bases: *PykeBot2.models.data_models.Payload*

Saves a list of teams and the name of the list

discord_extended_str()

discord_str()

extended_str()

name: str

teams: List[*PykeBot2.models.data_models.Team*]

class PykeBot2.models.data_models.**TeamListList**(*team_lists*)

Bases: *PykeBot2.models.data_models.Payload*

Saves a list of TeamList objects

discord_extended_str()

discord_str()

extended_str()

team_lists: List[*PykeBot2.models.data_models.TeamList*]

PykeBot2.models.errors module

exception PykeBot2.models.errors.**InvalidForwardToError**

Bases: Exception

Description Raised when the forward_to of a query could not be matched

exception PykeBot2.models.errors.**InvalidNextStepError**

Bases: Exception

Description Raised when the next_step of a query could not be matched

exception PykeBot2.models.errors.**NotFoundResponseError**

Bases: Exception

Description Raised when stalker gets Error Code 404 Not found for a request

exception PykeBot2.models.errors.**PayloadCreationError**

Bases: Exception

Description Raised when attempting to create a payload object and not a subclass

exception PykeBot2.models.errors.**ServerErrorResponseError**

Bases: Exception

Description Raised when a stalker gets Error Code 5xx for a request

exception PykeBot2.models.errors.**TokenLoadingError**

Bases: Exception

Description Raised when load_token is unable to load a token

PykeBot2.models.lookup_tables module

Contains various lookup tables :author: Jonathan Decker

PykeBot2.models.lookup_tables.**all_flags_lookup = {'f', 'file', 'group', 'no-api', 'r', 'rank', 'ranks'}**

lookup tables for query parameters

PykeBot2.models.lookup_tables.**battlefy_base_url = 'battlefy.com'**

help message returned when calling a help command

PykeBot2.models.lookup_tables.**debug_flag = False**

Version number TODO: Keep version number up to date!

PykeBot2.models.lookup_tables.**forward_to_lookup = {'backend', 'discord', 'frontend'}**

lookup tables for websites and website key words

PykeBot2.models.lookup_tables.**help_message = "Welcome to PykeBot2!\n\nEvery command has the pattern:\n.pb command [flags] [data]\nwhere valid commands are 'stalk' and 'help'.\n\nstalk takes a valid url to a tournament as data and stalks its teams and players.\nSupported for stalking are Prime League, Toornament, Summoners Inn and Battlefy.\n\nstalk accepts the flags:\n'rank', for adding ranks to each player via op.gg and\n'file', for output as file instead of as chat messages\n\nFor example: .pb stalk rank file <url>\nwould return teams and players for the given <url> with ranks as a file.\n\nhelp returns this message and ignores any flags or data.\nFor further information on PykeBot2 see:\nhttps://github.com/Twalord/PykeBot2\n\nAdditional flags:\n'group' to force prime league group stalker instead of season stalker.\n'no-api' to force using the HTML scraper instead of an api."**

Uniliga Seitenwahl rules, as we tend to forget or confuse them

PykeBot2.models.lookup_tables.**last_updated = '01.05.2022'**

lookup tables for commands and flags

PykeBot2.models.lookup_tables.**uniliga_seitenwahl_rules = '§ 3.2.2 Best-of-2 \nJedes Team spielt je einmal auf der linken und auf der rechten Seite.\nDas auf Toornament zuerst genannte Team darf dabei im ersten Spiel die Seite wählen.'**

lookup tables for rank_stalker

PykeBot2.models.query module

Query is the main object used for carrying information between different submodules. Normally a command from the user interface is represented by a single Query, but in some cases additional Queries carrying Messages might be created.

author Jonathan Decker

class PykeBot2.models.query.**Query**(*context_type: str*, *forward_to: str*, *next_step: str*, *raw_command: str = ''*, *discord_channel: None.Message.channel = None*, *data: str = ''*, *output_message: str = ''*, *flags: Set[str] = None*, *payload:* PykeBot2.models.data_models.Payload *= None*)

Bases: `object`

Description Class for any type of data that is sent around via Queues.

Should mainly be created by ingoing interfaces. While processing the query the fields should be further filled out and updated using update query function.

context_type: str

data: str

discord_channel: None.Message.channel

flags: Set[str]

forward_to: str

next_step: str

output_message: str

payload: *PykeBot2.models.data_models.Payload*

raw_command: str

update_query(*forward_to: str*, *next_step: str*, *data: Optional[str] = None*, *flags: Optional[Set[str]] = None*, *output_message: Optional[str] = None*, *payload: Optional[*PykeBot2.models.data_models.Payload*] = None*)

Description Updates query information, should be used instead of directly modifying attributes as some checks are

in place to validate the new values. :param forward_to: The next sub module that needs to handle the query. Used by the query forwarder. :type forward_to: str :param next_step: The next step in processing the command read by the submodule receiving the query. :type next_step: str :param data: Input data from the command, usually urls for stalking. :type data: str :param flags: A set of strings which sets additional options like output as file or stalk ranks. :type flags: Set[str] :param output_message: :type output_message: :param payload: Data model that carries the data produced by the backend, a message or an error message. :type payload: Payload :return: None :rtype: None

Module contents

PykeBot2.utils package

Submodules

PykeBot2.utils.config_master module

PykeBot2.utils.config_master.**initialize_config_parser**()

PykeBot2.utils.config_master.**reload_config_file**()

PykeBot2.utils.config_master.**restore_defaults**()

PykeBot2.utils.pb_logger module

Offers logging utilities for the entire program. Should be imported in every file as

"logger = logging.getLogger("pb_logger")"

author Jonathan Decker

PykeBot2.utils.pb_logger.**setup_logger**(*logger_name: str = 'pb_logger'*, *console_level=20*, *log_file_level=10*, *logs_to_keep: int = 20*, *create_log_files: bool = True*, *path_to_logs: Optional[pathlib.Path] = None*) → None

Description Sets up a logger with formatting, log file creation, settable console and log file logging levels as well as automatic deletion of old log files.

Parameters

- **logger_name** (*str*) – Name of the logger profile, standard is pb_logger, this should not be changed except for testing
- **console_level** (*logging level*) – logging level on console, standard is INFO
- **log_file_level** (*logging level*) – logging level in log file, standard is DEBUG
- **logs_to_keep** (*int*) – number of log files to keep before deleting the oldest one, standard is 20
- **create_log_files** (*bool*) – Whether to create log files or only log to console
- **path_to_logs** (*pathlib.PATH*) – path to the directory for saving the logs, standard is current working directory / logs

Returns None, but the logger may now be accessed via logging.getLogger(logger_name), standard is pb_logger

Return type None

PykeBot2.utils.token_loader module

Offers utility for loading API Tokens from a file or environmental variable.

author Jonathan Decker

PykeBot2.utils.token_loader.**load_token**(*name: str*, *try_env: bool = True*, *hide_token: bool = True*, *path_to_token: Optional[pathlib.Path] = None*) → str

Description Loads the specified token, trying either the given path or the cwd and if set the env variables

Parameters

- **name** (*str*) – Name of the token, as in the file_name or the name of the env variable
- **try_env** (*bool*) – Sets whether environmental variables should be checked if the token is not in the given path, standard is True
- **hide_token** (*bool*) – Sets whether the token should be hidden from logging, standard is True
- **path_to_token** (*pathlib.Path*) – Path to the folder containing the token, should not include the token itself, standard is the cwd

Returns The token as in the first line of the token file or the value of the env variable of the same name

Return type str

Raises TokenLoadingError

Module contents

1.1.2 Submodules

1.1.3 PykeBot2.event_loop_master module

author Jonathan Decker

async PykeBot2.event_loop_master.**dump_que**(*queue: asyncio.queues.Queue*)

Description Coroutine to dump the contents of a Queue into the log and clear the Queue. Only used for debugging.

Parameters **queue** (*asyncio.Queue*) – The queue to dump.

Returns None

Return type None

async PykeBot2.event_loop_master.**query_forwarder**(*forward_queue: asyncio.queues.Queue*, *sub_module_queues: {<class 'str'>: <class 'asyncio.queues.Queue'>}*)

Description Coroutine that manages the forward_queue and reads the forward_to field of incoming queries.

Based on the value of the field the query is submitted to the next Queue. :param forward_queue: The forward_queue that is awaited by this Coroutine. :type forward_queue: asyncio.Queue :param sub_module_queues:

A dictionary that maps the short names of each sub module to its Queue. :type sub_module_queues: {str: asyncio.Queue} :return: None :rtype: None

PykeBot2.event_loop_master.**run_main_loop**()

Description The main loop of the program. Uses asyncio event loop and ensures all main Coroutines.

Further all Queue objects are created here as they need to be present when starting the Coroutines. :return: None :rtype: None

async PykeBot2.event_loop_master.**sample_worker**(*num: int*)

Description Example for coroutine functionality, does not do any meaningful work.

Parameters **num** (*int*) – Number only used to track the worker in logs.

Returns None

Return type None

1.1.4 PykeBot2.gecko_manager module

Offers utility to open a firefox webdriver. Geckodriver should be in PATH for this.

author Jonathan Decker

PykeBot2.gecko_manager.**open_session**(*headless=True*) → selenium.webdriver.firefox.webdriver.WebDriver

Description Opens a Selenium Firefox web session.

Attempts to find geckodriver from PATH, if it fails uses a static path. :param headless: Set whether the web session should be headless or not. :type headless: bool :return: A new firefox webdriver. :rtype: selenium.webdriver

PykeBot2.gecko_manager.**quit_session**(*driver: <module 'selenium.webdriver' from '/home/docs/checkouts/readthedocs.org/user_builds/pykebot2/envs/stable/lib/python3.7/site-packages/selenium/webdriver/__init__.py'>*)

1.1.5 PykeBot2.main module

Main file used to start PykeBot2.

author Jonathan Decker

PykeBot2.main.**start**()

1.1.6 Module contents

Discord Bot for collecting information on league of legends players in tournaments.

CHAPTER

TWO

INDICES AND TABLES

- genindex
- modindex
- search

PYTHON MODULE INDEX

p

INDEX

E

F

G

H

I

L

M

N

O

P

Q

R

S

T

U

W