SpacePY-X Documentation

Release 0.1

Andrew Shapton

Jul 03, 2020

Contents

Installation

To install via `pip`, use:

```
pip install  spacePY-X
```

Example Usage

```
pip install spacePY-X

import spacePY-X

# Example usage:
rocket = spacexpython.rockets.falcon1()
print(rocket)
```

Source: NASA Images [Ref1]

CHAPTER 1

Capsule Information

This group of API calls will enable the retrieval of capsule data. ALL capsule calls can be given a set of parameters, with which to modify the response. Like all functions in this module, the API parameters must be given as a JSON payload such as can be seen *here*.

1.1 All Capsules

```
capsules = spacexpython.capsules.capsules(parameters,timeOut)
print(capsule)
```

Parameters:

Name	Purpose	Mandatory	Default
parameters	JSON list of URL qualifiers in the form {“status”:”active”,”limit”:3 ……etc	N	
timeOut	Number of seconds to wait until a timeout	N	1

Valid parameters

1.2 Specific Capsule

```
capsule = spacexpython.capsules.one(capsule_id,parameters,timeOut)
print(capsule)
```

Parameters:

Name	Purpose	Mandatory	Default
capsule_id	ID of the capsule	Y	
parameters	JSON list of URL qualifiers in the form {“status”:”active”,”limit”:3 ……etc	N	
timeOut	Number of seconds to wait until a timeout	N	1

Valid parameters

1.3 Upcoming Capsules

```
upcoming_capsules = spacexpython.capsules.upcoming(parameters,timeOut)
print(upcoming_capsules)
```

Parameters:

Name	Purpose	Mandatory	Default
parameters	JSON list of URL qualifiers in the form {“status”:”active”,”limit”:3 ……etc	N	
timeOut	Number of seconds to wait until a timeout	N	1

Valid parameters

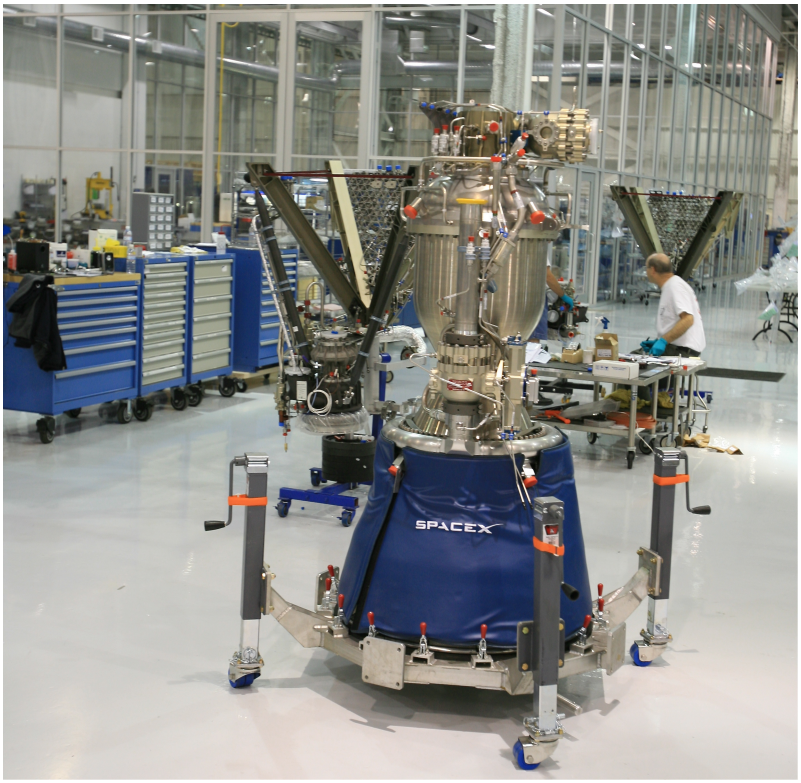
1.4 Past Capsules

```
past_capsules = spacexpython.capsules.past(parameters,timeOut)
print(past_capsules)
```

Parameters:

Name	Purpose	Mandatory	Default
parameters	JSON list of URL qualifiers in the form {“status”:”active”,”limit”:3 ……etc	N	
timeOut	Number of seconds to wait until a timeout	N	1

Valid parameters



Source: [Ref2]

CHAPTER 2

Core Information

This group of API calls will enable the retrieval of core data. In SpaceX terms, a "core" is defined as a rocket motor. ALL core calls can be given a set of parameters, with which to modify the response. Like all functions in this module, the API parameters must be given as a JSON payload such as can be seen *here*.

2.1 All Cores

```
cores = spacexpython.cores.cores(parameters,timeOut)
print(core)
```

Parameters:

Name	Purpose	Mandatory	Default
parameters	JSON list of URL qualifiers in the form {"status":"active","limit":3 .....etc	N	
timeOut	Number of seconds to wait until a timeout	N	1

Valid parameters

2.2 Specific Core

```
core = spacexpython.cores.one(core_id,parameters,timeOut)
print(core)
```

Parameters:

Name	Purpose	Mandatory	Default
core_id	ID of the core	Y	
parameters	JSON list of URL qualifiers in the form {“status”:”active”,”limit”:3 ……etc	N	
timeOut	Number of seconds to wait until a timeout	N	1

Valid parameters

2.3 Upcoming Cores

```
upcoming_cores = spacexpython.cores.upcoming(parameters,timeOut)
print(upcoming_cores)
```

Parameters:

Name	Purpose	Mandatory	Default
parameters	JSON list of URL qualifiers in the form {“status”:”active”,”limit”:3 ……etc	N	
timeOut	Number of seconds to wait until a timeout	N	1

Valid parameters

2.4 Past Cores

```
past_cores = spacexpython.cores.past(parameters,timeOut)
print(past_cores)
```

Parameters:

Name	Purpose	Mandatory	Default
parameters	JSON list of URL qualifiers in the form {“status”:”active”,”limit”:3 ……etc	N	
timeOut	Number of seconds to wait until a timeout	N	1

Valid parameters

Source: [Ref15]

CHAPTER 3

Crew Information

Attention: This is `experimental` in version 1.1.1a1

This is not complete in the API, therefore the wrapper will return no data until there is a complete implementation.

3.1 Crew Information

```
crew = spacexpython.crew.crew(timeOut)
print(crew)
```

Parameter:

Name	Purpose	Mandatory	Default
timeOut	Number of seconds to wait until a timeout	N	1

Source: NASA [Ref3]

CHAPTER 4

Dragon Information

This group of API calls will enable the retrieval of Dragon data. ALL Dragon calls can be given a set of parameters, with which to modify the response. Like all functions in this module, the API parameters must be given as a JSON payload such as can be seen *here*.

4.1 All Dragons

```
dragons = spacexpython.dragons.dragons(parameters,timeOut)
print(dragon)
```

Parameters:

Name	Purpose	Mandatory	Default
parameters	JSON list of URL qualifiers in the form {“status”:”active”,”limit”:3 ……etc	N	
timeOut	Number of seconds to wait until a timeout	N	1

Valid parameters

4.2 Specific Dragon

```
dragon = spacexpython.dragons.one(dragon_id,parameters,timeOut)
print(dragon)
```

Parameters:

Name	Purpose	Mandatory	Default
dragon_id	ID of the dragon capsule	Y	
parameters	JSON list of URL qualifiers in the form {“status”:”active”,”limit”:3 ..... etc	N	
timeOut	Number of seconds to wait until a timeout	N	1

Valid parameters

Source: [Ref4]

CHAPTER 5

History Information

This group of API calls will enable the retrieval of history data. History data is classed as significant steps forward in SpaceX's journey toward commercial spaceflight ALL history calls can be given a set of parameters, with which to modify the response. Like all functions in this module, the API parameters must be given as a JSON payload such as can be seen *here*.

5.1 All historic events

```
history = spacexpython.history.history(parameters,timeOut)
print(history)
```

Parameters:

Name	Purpose	Mandatory	Default
parameters	JSON list of URL qualifiers in the form {"status":"active","limit":3 ..... etc	N	
timeOut	Number of seconds to wait until a timeout	N	1

Valid parameters

5.2 Specific Historic event

```
event = spacexpython.history.one(event_id,parameters,timeOut)
print(event)
```

Parameters:

Name	Purpose	Mandatory	Default
event_id	ID of the historic event	Y	
parameters	JSON list of URL qualifiers in the form {“status”:”active”,”limit”:3 ..... etc	N	
timeOut	Number of seconds to wait until a timeout	N	1

Valid parameters

Source: [Ref5] Source Bruno Sanchez-Andrade Nuño from Washington, DC, USA‘

CHAPTER 6

Company and API Information

This group of API calls allows the retrieval of information regarding :

- SpaceX corporate information
- API Information
- Client Information
- Application Information

Like all functions in this module, the API parameters must be given as a JSON payload such as can be seen *here*.

6.1 Company Information

```
info = spacexpython.info.company(timeOut)
print(info)
```

Parameter:

Name	Purpose	Mandatory	Default
timeOut	Number of seconds to wait until a timeout	N	1

More Details

6.2 API Details

```
api = spacexpython.info.api(timeOut)
print(api)
```

Parameter:

Name	Purpose	Mandatory	Default
timeOut	Number of seconds to wait until a timeout	N	1

More Details

6.3 Client Information

This function interrogates the known list of clients (wrappers) in many different languages, with many options for filtering.

```
clients = spacexpython.info.clients(parameters,timeOut)
print(clients)
```

Parameters:

Name	Purpose	Mandatory	Default
parameters	JSON list of filters in the form {“status”:”active”,”limit”:3 ..... etc	N	
timeOut	Number of seconds to wait until a timeout	N	1

More Details

6.4 Application Information

This function will return information aout all, or specific applications from the known list of applications (wrappers).

```
applications = spacexpython.info.apps(parameters,timeOut)
print(applications)
```

Parameters:

Name	Purpose	Mandatory	Default
parameters	JSON list of filters in the form {“status”:”active”,”limit”:3 ..... etc	N	
timeOut	Number of seconds to wait until a timeout	N	1

More Details

Source: SpaceX [Ref6]

CHAPTER 7

Landing Pad Information

This group of API calls will enable the retrieval of data about the various landing pads that SpaceX uses. ALL these calls can be given a set of parameters, with which to modify the response. Like all functions in this module, the API parameters must be given as a JSON payload such as can be seen *here*.

7.1 All Landing Pads

```
landingpads = spacexpython.landingpads.landingpads(parameters,timeOut)
print(landingpads)
```

Parameters:

Name	Purpose	Mandatory	Default
parameters	JSON list of URL qualifiers in the form {“status”:”active”,”limit”:3 ..... etc	N	
timeOut	Number of seconds to wait until a timeout	N	1

Valid parameters

7.2 Specific Landing Pad

```
landingpad = spacexpython.landingpads.one(landingpad_id,parameters,timeOut)
print(landingpad)
```

Parameters:

Name	Purpose	Mandatory	Default
landingpad_id	ID of the landingpad	Y	
parameters	JSON list of URL qualifiers in the form {“status”:”active”,”limit”:3 . . . . . etc	N	
timeOut	Number of seconds to wait until a timeout	N	1

Valid parameters

Source: SpaceX [Ref7]

CHAPTER 8

Launch Information

This group of API calls will enable the retrieval of Launch data. ALL Launch calls can be given a set of parameters, with which to modify the response. Like all functions in this module, the API parameters must be given as a JSON payload such as can be seen *here*.

8.1 All Launches

```
launches = spacexpython.launches.launches(parameters,timeOut)
print(launches)
```

Parameters:

Name	Purpose	Mandatory	Default
parameters	JSON list of URL qualifiers in the form {“status”:”active”,”limit”:3 ……etc	N	
timeOut	Number of seconds to wait until a timeout	N	1

Valid parameters

8.2 Specific Launch

Important: As of version v1.1.0.alpha4, this feature is not implemented. It will be implemented in a future release.

```
launch = spacexpython.launches.one(launch_id,parameters,timeOut)
print(Launch)
```

Parameters:

Name	Purpose	Mandatory	Default
launch_id	ID of the Launch	Y	
parameters	JSON list of URL qualifiers in the form {“status”:”active”,”limit”:3 ……etc	N	
timeOut	Number of seconds to wait until a timeout	N	1

Valid parameters

8.3 Upcoming Launches

```
upcoming_launches = spacexpython.launches.upcoming(parameters,timeOut)
print(upcoming_launches)
```

Parameters:

Name	Purpose	Mandatory	Default
parameters	JSON list of URL qualifiers in the form {“status”:”active”,”limit”:3 ……etc	N	
timeOut	Number of seconds to wait until a timeout	N	1

Valid parameters

8.4 Next Launch

```
next_launch = spacexpython.launches.nextlaunch(timeOut)
print(next_launch)
```

Parameters:

Name	Purpose	Mandatory	Default
timeOut	Number of seconds to wait until a timeout	N	1

Valid parameters

8.5 Latest Launch

```
latest_launch = spacexpython.launches.latest(timeOut)
print(latest_launch)
```

Parameters:

Name	Purpose	Mandatory	Default
timeOut	Number of seconds to wait until a timeout	N	1

Valid parameters

8.6 Past Launches

Important: As of version v1.1.0.alpha4, this feature is not implemented. It will be implemented in a future release.

```
past_launches = spacexpython.launches.past(parameters,timeOut)
print(past_launches)
```

Parameters:

Name	Purpose	Mandatory	Default
parameters	JSON list of URL qualifiers in the form {“status”:”active”,”limit”:3 . . . . . etc	N	
timeOut	Number of seconds to wait until a timeout	N	1

Valid parameters

Source: NASA/Kim Shiflett [Ref13]

CHAPTER 9

Launch Pad Information

This group of API calls will enable the retrieval of data about the various launch pads that SpaceX uses. ALL these calls can be given a set of parameters, with which to modify the response. Like all functions in this module, the API parameters must be given as a JSON payload such as can be seen *here*.

9.1 All Launch Pads

```
launchpads = spacexpython.launchpads.launchpads(parameters,timeOut)
print(launchpads)
```

Parameters:

Name	Purpose	Mandatory	Default
parameters	JSON list of URL qualifiers in the form {“status”:”active”,”limit”:3 ……etc	N	
timeOut	Number of seconds to wait until a timeout	N	1

Valid parameters

9.2 Specific Launch Pad

```
launchpad = spacexpython.launchpads.one(launchpad_id,parameters,timeOut)
print(launchpad)
```

Parameters:

Name	Purpose	Mandatory	Default
launchpad_id	ID of the launchpad	Y	
parameters	JSON list of URL qualifiers in the form {“status”:”active”,”limit”:3 ..... etc	N	
timeOut	Number of seconds to wait until a timeout	N	1

Valid parameters

Source: Collectspace [Ref8]

CHAPTER 10

Mission Information

This group of API calls will enable the retrieval of data about the various missions that SpaceX has been involved in. ALL these calls can be given a set of parameters, with which to modify the response. Like all functions in this module, the API parameters must be given as a JSON payload such as can be seen *here*.

10.1 All Missions

```
missions = spacexpython.missions.missions(parameters,timeOut)
print(missions)
```

Parameters:

Name	Purpose	Mandatory	Default
parameters	JSON list of URL qualifiers in the form {“status”:”active”,”limit”:3 ……etc	N	
timeOut	Number of seconds to wait until a timeout	N	1

Valid parameters

10.2 Specific Mission

```
mission = spacexpython.missions.one(mission_id,parameters,timeOut)
print(mission)
```

Parameters:

Name	Purpose	Mandatory	Default
mission_id	ID of the mission	Y	
parameters	JSON list of URL qualifiers in the form {“status”:”active”,”limit”:3 ……etc	N	
timeOut	Number of seconds to wait until a timeout	N	1

Valid parameters

Source: NASA [Ref9]

CHAPTER 11

Payload Information

This group of API calls will enable the retrieval of data about the various payloads that SpaceX has (or will be) responsible for shipping. ALL these calls can be given a set of parameters, with which to modify the response. Like all functions in this module, the API parameters must be given as a JSON payload such as can be seen *here*.

11.1 All Payloads

```
payloads = spacexpython.payloads.payloads(parameters,timeOut)
print(payloads)
```

Parameters:

Name	Purpose	Mandatory	Default
parameters	JSON list of URL qualifiers in the form {“status”:”active”,”limit”:3 ……etc	N	
timeOut	Number of seconds to wait until a timeout	N	1

Valid parameters

11.2 Specific Payload

```
payload = spacexpython.payloads.one(payload_id,parameters,timeOut)
print(payload)
```

Parameters:

Name	Purpose	Mandatory	Default
payload_id	ID of the payload	Y	
parameters	JSON list of URL qualifiers in the form {“status”:”active”,”limit”:3 ……etc	N	
timeOut	Number of seconds to wait until a timeout	N	1

Valid parameters

Source: Falcon Heavy Test Mission [Ref11]

CHAPTER 12

Roadster Information

This function will return the current orbital parameters of Elon Musk's Tesla Roadster which was launched in February 2018 aboard the first test flight of a Falcon Heavy Rocket.

12.1 Roadster Data

```
roadster = spacexpython.roadster.roadster(timeOut)
print(roadster)
```

Parameter:

Name	Purpose	Mandatory	Default
timeOut	Number of seconds to wait until a timeout	N	1

More Details

Source: SpaceX [Ref10]

CHAPTER 13

Rocket Information

This group of API calls will enable the retrieval of data about the rockets that SpaceX has used over its' launch timeframe. ALL these calls can be given a set of parameters, with which to modify the response. Like all functions in this module, the API parameters must be given as a JSON payload such as can be seen *here*.

13.1 All Rockets

```
rockets = spacexpython.rockets.rockets(parameters,timeOut)
print(rockets)
```

Parameters:

Name	Purpose	Mandatory	Default
parameters	JSON list of URL qualifiers in the form {"status":"active","limit":3 ..... etc	N	
timeOut	Number of seconds to wait until a timeout	N	1

Valid parameters

13.2 Specific Rocket

```
rocket = spacexpython.rockets.rocket(rocket_id,parameters,timeOut)
print(rocket)
```

Note: The *rocket_id* parameter is one of :

Value	Meaning
falcon1	Falcon 1 Rocket
falcon9	Falcon 1 Rocket
falconheavy	Falcon Heavy Rocket
bfr	Big Falcon Rocket
starship	Starship

Parameters:

Name	Purpose	Mandatory	Default
rocket_id	ID of the rocket	Y	
parameters	JSON list of URL qualifiers in the form {"status":"active","limit":3 ..... etc	N	
timeOut	Number of seconds to wait until a timeout	N	1

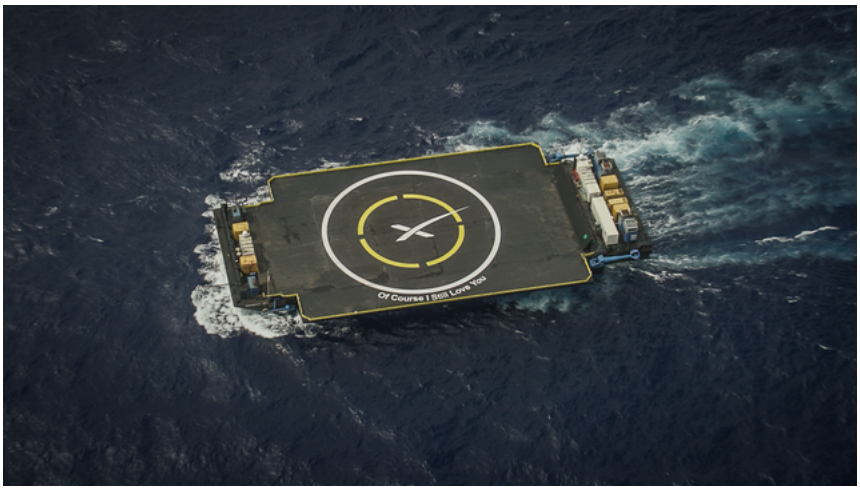
An additional method of acquiring information about a specific rocket would be to use the rocket-specific functions:

```
falcon1     = spacexpython.rockets.falcon1(parameters, timeOut)
falcon9     = spacexpython.rockets.falcon9(parameters, timeOut)
falconheavy = spacexpython.rockets.falconheavy(parameters, timeOut)
bfr         = spacexpython.rockets.bfr(parameters, timeOut)
starship    = spacexpython.rockets.starship(parameters, timeOut)
```

Valid parameters

Note: Beginning with version 1.1.1.a1, any references to "bfr" will cause the information from the "starship" function to be returned. This is because Elon Musk, on the 20th November 2018 renamed Big Falcon Rocket to Starship. [RefBFR]

Additionally, as of version 1.1.2, the function and references to "bfr" are considered to be deprecated.



Source: SpaceX [Ref12]

CHAPTER 14

Ship Information

This group of API calls will enable the retrieval of data about the ships that SpaceX has used over its' operational timeframe. These include autonomous drone ships, barges, recovery and transport ships. ALL these calls can be given a set of parameters, with which to modify the response. Like all functions in this module, the API parameters must be given as a JSON payload such as can be seen *here*.

14.1 All Ships

```
ships = spacexpython.ships.ships(parameters,timeOut)
print(ships)
```

Parameters:

Name	Purpose	Mandatory	Default
parameters	JSON list of URL qualifiers in the form {"status":"active","limit":3 ..... etc	N	
timeOut	Number of seconds to wait until a timeout	N	1

Valid parameters

14.2 Specific Ship

```
ship = spacexpython.ships.ship(ship_id,parameters,timeOut)
print(ship)
```

Parameters:

Name	Purpose	Mandatory	Default
ship_id	ID of the ship	Y	
parameters	JSON list of URL qualifiers in the form {“status”:”active”,”limit”:3 ..... etc	N	
timeOut	Number of seconds to wait until a timeout	N	1

Valid parameters

CHAPTER 15

Application function Parameters

Each parameter is an incremental filter, thus each filter will narrow down the return set.

The value should be expressed as a valid JSON payload, such as the following:

```
{
    "Creators":"["Albert Gallileo","Sergey Zuckerberg"]",
    "Platforms":"iOS"
}
```

An example of this would be:

```
parameters = {"Creators":"["Albert Gallileo",
                "Sergey Zuckerberg"]",
                "Platforms":"iOS"}                      # JSON payload indicating␣
↪only iOS applications clients
                                                        # written by Albert Gallileo␣
↪or Sergey Zuckerberg should be returned
timeOut = 2                                             # wait 2 seconds before␣
↪recording a timeout
applications = spacexpython.info.apps(parameters,timeOut)
print(applications)
```

The JSON-format parameter list consists of one or more of :

Key	Meaning	Type	Example
Name	Application name	str	SpaceX-GraphQL
Links	Links(s) to the application name		NOT FILTERABLE
Types	Application type e.g. API, Website	dict	[“Website”, “App”, “Bot”, “API”]
Platforms	The platform(s) on which the application resides	dict	**[“iOS”, “Android”, “Discord”, “iOS”, “Android”, “D** “Web”, “API”]
Creators	The creator(s) of the application	dict	[“Noah Zyrgunski”]
CreatorsLinks	Link(s) to creators detailed above		NOT FILTERABLE
Repos	Nature(s) of Repo(s)	dict	[“Github”, “N/A”, <other text>]
ReposLinks	Link(s) to repos detailed above		NOT FILTERABLE
More	Additional details		NOT FILTERABLE
MoreLinks	Link(s) to additional details as above		NOT FILTERABLE

The following is also valid syntax, and will return a complete list of applications:

```
parameters = ''                                          # ALL applications will be␣
↪returned
timeOut = 3                                              # wait 3 seconds before␣
↪recording a timeout
allapps = spacexpython.info.apps(parameters,timeOut)
print(allapps)
```

Note: This feature is new as of v1.1.2

CHAPTER 16

Client function Parameters

Each parameter is an incremental filter, thus each filter will narrow down the return set.

The value should be expressed as a valid JSON payload, such as the following:

```
{
    "Name":"SpacePY-X",
    "Language":"Python"
}
```

An example of this would be:

```
parameters = {"Creators":"Harry Schroedinger",
                "Language":"Python"}                    # JSON payload indicating␣
↪only Python clients
                                                        # written by Harry␣
↪Schroedinger should be returned
timeOut = 2                                             # wait 2 seconds before␣
↪recording a timeout
clients = spacexpython.info.clients(parameters,timeOut)
print(clients)
```

The JSON-format parameter list consists of one or more of :

Key	Meaning	Type	Example
Name	Client name	str	SpaceX-GraphQL
Languages	Language(s) in which the client is written	dict	[“Python”, “Rust”]
Creators	List of creator(s)	dict	[“William deAngelis”]
Repos	Nature(s) of Repo	dict	[“Github”, “NPM”, “Website”]
ReposLinks	Link(s) to repos detailled above	NOT FILTERABLE	

The following is also valid syntax, and will return a complete list of clients:

```
parameters = ''                                          # ALL clients will be returned
timeOut = 3                                              # wait 3 seconds before␣
↪recording a timeout
clients = spacexpython.info.clients(parameters,timeOut)
print(clients)
```

Note: This feature is new as of v1.1.2

CHAPTER 17

Exceptions

Should an error occur anywhere in the call to a function (whether that be in the wrapper or the REST API itself), an exception will be raised.

All normal Python exceptions exist that can be trapped, however, the wrapper implements 2 new exceptions:

```
SpaceXReadTimeOut
```

This exception occurs when the API times out for any reason. It can be trapped, and possibly a retry or other action performed as appropriate, for example:

```
try:
    capsules_data = keyOrder(alphaOrder(spacexpython.capsules.capsules('',1)),
↪'capsule_serial')
except spacexpython.utils.SpaceXReadTimeOut:
    print("Failure on retrieval of capsule information")
```

..code-block:: python

SpaceXParameterError

This exception occurs when the the paramaters for a wrapper call (and ultimately to the API itself) do not meet the type specifications set out in the core API definition:

```
try:
    coresP_data = alphaOrder(spacexpython.cores.cores('{"core_serial":"B1037","block":
↪"true"}'))
except spacexpython.utils.SpaceXParameterError:
    print("Incorrect parameter")
```

This error is due to the `block` parameter in this instance showing `true` when the parameter is defined as an `integer`.

A further example :

```
try:
    coresP_data = alphaOrder(spacexpython.cores.cores('{"core_serial":"B1037",
↪"desired-block":"4"}'))
except spacexpython.utils.SpaceXParameterError:
    print("Incorrect parameter")
```

This error is due to the `block` parameter in this instance not being a valid parameter.

CHAPTER 18

JSON Parameters

Most functions within thus wrapper will take a parameter (typically called “parameters”).

The value is a modifier/qualifier to the function call that is being made, and will affect the results of that function.

The value should be expressed as a valid JSON payload, such as the following:

```
{
    "status":"active",
    "limit":"3"
}
```

An example of this would be:

```
parameters = {"status":"active", "limit":"3"}     # JSON payload indicating only 3␣
↪active
                                                  # capsules should be returned
timeOut = 2                                       # wait 2 seaconds before recording a␣
↪timeout
capsules = spacexpython.capsules.capsules(parameters,timeOut)
print(capsule)
```

Note: As of version 1.1.1a2, initial validation has been introduced. This should NOT be considered production ready yet.

Up to and including version 1.0.0.alpha4, no parameter validation took place, therefore incorrect parameters would produce unknown and unpredictable effects. Please check all spelling and validity of parameters prior to use.

CHAPTER 19

spacexpython

19.1 spacexpython package

CHAPTER 20

Simple and Easy API Wrapper for r-spacex/SpaceX-API

20.1 Documentation

This API Wrapper aims to provide a simple and easy way to use the SpaceX-API in Python projects.

Installation: Note that this supports Python 3 *ONLY*

To install via `pip` use: `pip install spacePY-X`

Authentication

No authentication is currently required by r-spacex/SpaceX-API to use this public API and thus is not required in this wrapper.

Rate Limiting

The API has a rate limit of 50 req/sec per IP address, if exceeded, a response of 429 will be given until the rate drops back below 50 req/sec.

Credits

Jake Meyer's r-spacex/SpaceX-API

Based on work by vinayphadnis

License MIT

Bibliography

[Ref1] iss058e027464 (March 4, 2019) — The uncrewed SpaceX Crew Dragon spacecraft is the first Commercial Crew vehicle to visit the International Space Station. Here it is pictured with its nose cone open revealing its docking mechanism while approaching the station's Harmony module. The Crew Dragon would automatically dock moments later to the international docking adapter attached to the forward end of Harmony.

[Ref2] By Steve Jurvetson from Menlo Park, USA - Flickr: Merlin Engine, CC BY 2.0 - One of nine Merlin engines used in the Falcon 9 booster.

[Ref15] The first two crew for SpaceX (Mission SpaceX Demo-2) planned for 30th November 2019. Crew members are Douglas G Hurley and Robert L Behnken (both of NASA)

[Ref3] Source The SpaceX Dragon space freighter approaches the International Space Station as both spacecraft were orbiting 265 miles above the Atlantic Ocean off the west coast of Namibia.

[Ref4] The second-stage Kestrel engine glows red-hot during Falcon 1's fourth launch and first successful orbital flight.

[Ref5] The company's headquarters, located in Hawthorne, California.

[Ref6] Falcon 9 First Stage Landing at LZ-1 on the CRS-11 mission

[Ref7] Falcon 9 and Dragon lift off from Launch Pad 39A for CRS-10

[Ref13] Taken: October 23rd 2009 - An aerial view of Space Launch Complex 40 on Cape Canaveral Air Force Station. The pad will be used to support the new Falcon rockets to be launched by Space Exploration Technologies, known as SpaceX. Camera Location

[Ref8] NASA's insignia for the tenth Commercial Resupply Services flight to the International Space Station, with "mousetronauts." Also includes the Stratospheric Aerosol and Gas Experiment (SAGE) and a Lightning Imaging Sensor, designated STP-H5.

[Ref9] This time-lapse animation shows NICER being extracted from the SpaceX Dragon trunk on June 11, 2017

[Ref11] Elon Musk's Tesla Roadster, with Earth in background. "Spaceman" mannequin wearing SpaceX Spacesuit in driving seat. Camera mounted on external boom.

[Ref10] Launch of Falcon 9 for mission CRS-12 14th August 2017

[RefBFR] BBC News Story

[Ref12] "Of Course I Still Love You" (Marmac 304) drone ship moving into position for the Sunday, June 28, 2015, SpaceX CRS-7 rocket launch and landing attempt.