petpy Documentation

Release 2.2.0

Aaron Schlegel

Jun 30, 2020

Contents

Petpy is an unofficial Pythonwrapper of the Petfinder API for interacting with Petfinder's database of animals and animal welfare organizations.

CHAPTER 1

Getting a Petfinder API and Secret Key

An account must first be created with Petfinder to receive an API and secret key. The API and secret key will be used to grant access to the Petfinder API, which lasts for 3600 seconds, or one hour. After the authentication period ends, you must re-authenticate with the Petfinder API.

CHAPTER 2

Installation

petpy is best installed through pip.

```
pip install petpy
```

For those of you who prefer it, the library can also be cloned or downloaded into a location of your choosing and then installed using the setup.py script per the following:

```
git clone git@github.com:aschleg/petpy.git
cd petpy
python setup.py install
```

CHAPTER 3

Requirements

- Python >= 3.6
- requests >= 2.18.4
- Although not strictly required to use `petpy`, the pandas library (>=0.22.0) is needed for returning the results as a DataFrame.

CHAPTER 4

Contents

4.1 API Reference

4.1.1 `Petfinder` – Petfinder API Wrapper

class petpy.api.**Petfinder**(*key*, *secret*)

The Petfinder class provides the wrapper for the Petfinder API. The API methods are listed below

Parameters

- **key** – API key received from Petfinder after creating a developer account.
- **secret** – Secret key received from Petfinder.

```
import petpy
pf = Petfinder(key=API_key, secret=API_secret)
```

4.1.2 Get Animal Types

Petfinder.**animal_types**([*types=None*])

Returns data on an animal type, or types available from the Petfinder API. This data includes the available type's coat names and colors, gender and other specific information relevant to the specified type(s). The animal type must be of 'dog', 'cat', 'rabbit', 'small-furry', 'horse', 'bird', 'scales-fins-other', 'barnyard'.

Parameters **types** – String representing a single animal type or a list or tuple of a collection of animal types. If not specified, all available breeds for each animal type is returned. The animal type must be of 'dog', 'cat', 'rabbit', 'small-furry', 'horse', 'bird', 'scales-fins-other', 'barnyard'.

Return type dict. Dictionary object representing JSON data returned from the Petfinder API.

```
# All animal types and their relevant data.
all_types = pf.animal_types()

# Returning data for a single animal type
dogs = pf.animal_types('dog')

# Getting multiple animal types at once
cat_dog_rabbit_types = pf.animal_types(['cat', 'dog', 'rabbit'])
```

4.1.3 Get Available Animal Breeds

`Petfinder.`**`breeds`**(*types[, return_df=False][, raw_results=False]*)

Returns breed names of specified animal type, or types.

Parameters

- **`types`** – String representing a single animal type or a list or tuple of a collection of animal types. If not specified, all available breeds for each animal type is returned. The animal type must be of 'dog', 'cat', 'rabbit', 'small-furry', 'horse', 'bird', 'scales-fins-other', 'barnyard'.
- **`return_df`** – If True, coerces results returned from the Petfinder API into a pandas DataFrame.
- **`raw_results`** – The PetFinder API `breeds` endpoint returns some extraneous data in its result set along with the breed names of the specified animal type(s). If `raw_results` is `False`, the method will return a cleaner JSON object result set with the extraneous data removed. This parameter can be set to `True` for those interested in retrieving the entire result set. If the parameter `return_df` is set to `True`, a pandas `DataFrame` will be returned regardless of the value specified for the `raw_result` parameter.

Return type dict or pandas DataFrame. If the parameter `return_df` is `False`, a dictionary object representing the JSON data returned from the Petfinder API is returned. If `return_df=True`, the resulting dictionary is coerced into a pandas DataFrame. Note if `return_df=True`, the parameter `raw_results` is overridden.

```
cat_breeds = pf.breeds('cat')
dog_breeds = pf.breeds('dog')

# All available breeds or multiple breeds can also be returned.
all_breeds = pf.breeds()
cat_dog_rabbit = pf.breeds(types=['cat', 'dog', 'rabbit'])

# The `breeds` method can also be set to coerce the returned JSON results into a␣
↪pandas DataFrame
# by setting the parameter `return_df = True`.

cat_breeds_df = pf.breeds('cat', return_df = True)
all_breeds_df = pf.breeds(return_df = True)
```

4.1.4 Find Listed Animals on Petfinder

Petfinder.**animals**(*[animal_id=None][, animal_type=None][, breed=None][, size=None][, gender=None][, age=None][, color=None][, coat=None][, status=None][, name=None][, organization_id=None][, location=None][, distance=None][, sort=None][, results_per_page=None][, pages=None][, return_df=False]*)

Returns adoptable animal data from Petfinder based on specified criteria.

Parameters

- **animal_id** – Integer or list or tuple of integers representing animal IDs obtained from Petfinder. When `animal_id` is specified, the other function parameters are overridden. If `animal_id` is not specified, a search of animals on Petfinder matching given criteria is performed.
- **animal_type** – String representing desired animal type to search. Must be one of 'dog', 'cat', 'rabbit', 'small-furry', 'horse', 'bird', 'scales-fins-other', or 'barnyard'.
- **breed** – String or tuple or list of strings of desired animal type breed to search. Available animal breeds in the Petfinder database can be found using the `breeds()` method.
- **size** – String or tuple or list of strings of desired animal sizes to return. The specified size(s) must be one of 'small', 'medium', 'large', or 'xlarge'.
- **gender** – String or tuple or list of strings representing animal genders to return. Must be of 'male', 'female', or 'unknown'.
- **age** – String or tuple or list of strings specifying animal age(s) to return from search. Must be of 'baby','young', 'adult', 'senior'.
- **color** – String representing specified animal 'color' to search. Colors for each available animal type in the Petfinder database can be found using the `animal_types()` method.
- **coat** – Desired coat(s) to return. Must be of 'short', 'medium', 'long', 'wire', 'hairless', or 'curly'.
- **status** – Animal status to filter search results. Must be one of 'adoptable', 'adopted', or 'found'.
- **name** – Searches for animal names matching or partially matching name.
- **organization_id** – Returns results for specified `organization_id`. Can be a str or a tuple or list of str representing multiple organizations.
- **location** – Returns results by specified location. Must be in the format 'city, state' for city-level results, 'latitude, longitude' for lat-long results, or 'postal code'.
- **distance** – Returns results within the distance of the specified location. If not given, defaults to 100 miles. Maximum distance range is 500 miles.
- **sort** – Sorts by specified attribute. Leading dashes represents a reverse-order sort. Must be one of 'recent', '-recent', 'distance', or '-distance'.
- **pages** – The number of pages of results to return. For example, if `pages=4` with the default `count` parameter (25), 100 results would be returned. The paged results are returned as a list.
- **good_with_cats** – Filters returned animal results to animals that are designated as good with cats. Must be a boolean value (True, False) or a value that can be coerced to a boolean (1, 0).

- **good_with_children** – Filters returned animal results to animals that are designated as good with children. Must be a boolean value (True, False) or a value that can be coerced to a boolean (1, 0).
- **good_with_dogs** – Filters returned animal results to animals that are designated as good with dogs. Must be a boolean value (True, False) or a value that can be coerced to a boolean (1, 0).
- **before_date** – Returns results with a *published_at* datetime before the specified time. Must be a string in the form of 'YYYY-MM-DD' or 'YYYY-MM-DD H:M:S' or a datetime object.
- **after_date** – Returns results with a *published_at* datetime after the specified time. Must be a string in the form of 'YYYY-MM-DD' or 'YYYY-MM-DD H:M:S' or a datetime object.
- **results_per_page** – Number of results to return per page. Defaults to 20 results and cannot exceed 100 results per page.
- **return_df** – If True, coerces results returned from the Petfinder API into a pandas DataFrame.

Return type dict or pandas DataFrame. Dictionary object representing the returned JSON object from the Petfinder API. If `return_df=True`, the results are returned as a pandas DataFrame.

```
# Getting first 20 results without any search criteria
animals = pf.animals()

# Extracting data on specific animals with animal_ids

animal_ids = []
for i in animals['animals'][0:3]:
   animal_ids.append(i['id'])

animal_data = pf.animals(animal_id=animal_ids)

# Returning a pandas DataFrame of the first 150 animal results
animals = pf.animals(results_per_page=50, pages=3, return_df=True)
```

4.1.5 Get Animal Welfare Organization Data

`Petfinder.`**`organizations`**(*[organization_id=None][, name=None][, location=None][, distance=None][, state=None][, country=None][, query=None][, sort=True][, results_per_page=None][, pages=None][, return_df=False]*)

Returns data on an animal welfare organization, or organizations, based on specified criteria.

Parameters

- **organization_id** – Returns results for specified `organization_id`. Can be a str or a tuple or list of str representing multiple organizations.
- **name** – Returns results matching or partially matching organization name.
- **location** – Returns results by specified location. Must be in the format 'city, state' for city-level results, 'latitude, longitude' for lat-long results, or 'postal code'.
- **distance** – Returns results within the distance of the specified location. If not given, defaults to 100 miles. Maximum distance range is 500 miles.

- **state** – Filters the results by the selected state. Must be a two-letter state code abbreviation of the state name, such as 'WA' for Washington or 'NY' for New York.
- **country** – Filters results to specified country. Must be a two-letter abbreviation of the country and is limited to the United States and Canada.
- **query** – Search matching and partially matching name, city or state.
- **sort** – Sorts by specified attribute. Leading dashes represents a reverse-order sort. Must be one of 'recent', '-recent', 'distance', or '-distance'.
- **count** – Number of results to return per page. Defaults to 20 results and cannot exceed 100 results per page.
- **pages** – The number of pages of results to return. For example, if `pages=4` with the default `count` parameter (25), 100 results would be returned. The paged results are returned as a list.
- **return_df** – If True, coerces results returned from the Petfinder API into a pandas DataFrame.

Return type dict or pandas DataFrame. Dictionary object representing the returned JSON object from the Petfinder API. If `return_df=True`, the results are returned as a pandas DataFrame.

```
# Return the first 1,000 animal welfare organizations as a pandas DataFrame
organizations = pf.organizations(results_per_page=100, pages=10, return_df=True)

# Get organizations in the state of Washington
wa_organizations = pf.organizations(state='WA')
```

4.2 API Exceptions

class petpy.exceptions.**PetfinderError**(*Exception*)
Base Exception class for Petfinder API Exception definitions.

class petpy.exceptions.**PetfinderInvalidCredentials**(*PetfinderError*)
Exception for handling invalid API and secret keys passed to the Petfinder class.

class petpy.exceptions.**PetfinderInsufficientAccess**(*PetfinderError*)
Exception for handling insufficient access errors when working with the Petfinder API. This exception is typically raised when the credentials supplied to the Petfinder API have expired and the connection to the API needs to be re-authenticated.

class petpy.exceptions.**PetfinderResourceNotFound**(*PetfinderError*)
Exception for handling unknown resource requests.

class petpy.exceptions.**PetfinderUnexpectedError**(*PetfinderError*)
Exception for handling unexpected errors from the Petfinder API. This error is generally the result of an unknown and unexpected error that occurs on the server-side of the Petfinder API when sending a request.

class petpy.exceptions.**PetfinderInvalidParameters**(*PetfinderError*)
Exception for handling invalid values passed to Petfinder API method parameters.

4.3 Version History

Changelog and version changes made with each release.

4.3.1 Version 2.2.0

- Support for Python 3.5 has been discontinued.
- The `animals()` method has been updated to include new parameters provided by Petfinder's `animal` endpoint. This parameters include: - `good_with_cats`: Boolean for filtering animals that are designated as good with cats. - `good_with_children`: Boolean for filtering animals that are designated as good with children. - `good_with_dogs`: Boolean for filtering animals that are designated as good with dogs. - `before_date`: Returns results published before the specified time. - `after_date`: Returns results published after the specified time.

4.3.2 Version 2.1.3

- `organization_id` parameter in the `animals` method should now only return animals from specified organization IDs.

4.3.3 Version 2.1.2

New release includes a bug fix and some additional changes for checking total usage against the Petfinder API.

- `animal_type` parameter used in the `animals()` endpoint has been corrected and should be working properly.
- New methods for checking the usage of the supplied API key against the limits defined by the Petfinder API have been implemented to better help warn users when they are approaching their API request limit.
  - If the `max_pages` parameter exceeds 10,000, a warning will be issued to give the user the opportunity to continue or limit their results to 10,000 pages.
  - When the API limits are exceeded, a `RuntimeError` will be issued.

Thank you to contributor ma755 for submitting the pull request.

4.3.4 Version 2.1.1

Small update release to fix the `distance` parameter logic when searching for pets using the `animals()` or `organizations()` methods. Thank you to contributor ljlevins for submitting the pull request.

4.3.5 Version 2.1.0

The `2.1.0` release of `petpy` implements several user-defined exceptions to help users debug any errors that may occur. Although the `petpy` library attempts to find any invalid passed parameter values and raise the appropriate error before the call to the Petfinder API is made to reduce the number of calls made to the API; however, some errors cannot be caught until after the API call is made. This update introduces these custom, user-defined exceptions for debugging error responses from the Petfinder API. For more information on the Petfinder API error definitions, please see the Petfinder API documentation.

The following is a list of the new user-defined exceptions.

- `PetfinderInvalidCredentials` - Raised when a user enters invalid API key and secret key credentials.
- **`PetfinderInsufficientAccess`**
  - Raised when a status code 403 occurs. This error would typically be raised when the current authentication token to the Petfinder API has expired and requires the connnection to be re-authenticated.

- **PetfinderResourceNotFound**
  - Raised when a status code 404 occurs.
- **PetfinderUnexpectedError**
  - Raised when a status code 500 occurs.
- **PetfinderInvalidParameters**
  - Raised when a 400 status code occurs. The exception will include the invalid parameters detected by the Petfinder API and include those parameters as part of the error output as a JSON object. For more information on this error, please see the Petfinder API error documentation.
  - Please note the *petpy* library will attempt to catch any invalid parameters before the API call is made to avoid extraneous issuage of the API, but if an invalid parameter does get through then this error should help provide the necessary information for users to debug any errors related to their chosen parameters.

The following other changes have been made in the `2.1.0` release.

- The `host` and `auth` attributes of the `Petfinder` class are now private (to the extent that Python allows private attributes, denoted with an underscore in front of the attribute).

4.3.6 Version 2.0.2

Minor bug fix release with following:

- `breeds()` method now correctly returns data when a single animal type is supplied.
- `animals()` method now properly displays the correct error message when the `distance` parameter is 0 <= distance <= 500.

4.3.7 Version 2.0.1

- Fixes the `animals()` and `organizations()` method to return all matching search results when the `pages` parameter is set to `None`.
- The resulting JSON (dictionary) from the `breeds()` method when the parameter `return_df=False` will now consistently have `types` as the key. Prior to this change, if the `breeds()` method was called with a single animal type, the resulting key name in the returned object would be named `type`, whereas if more than one animal type is specified the key name would be `types`.
- The `distance` parameter for the `animals()` and `organizations()` parameters will now raise a `ValueError` if it is less than 0.

4.3.8 Version 2.0.0

New major release coinciding with the release of v2.0 of the Petfinder API! The legacy version of the Petfinder API, v1.0, will be retired in January 2020, therefore, the `petpy` library has been updated almost from the ground up to be compatible as possible with the new version of the Petfinder API! The new version of the Petfinder API is a huge improvement over the legacy version, with many changes and additions to the design of the API itself. As such, several methods from earlier releases of `petpy` that wrapped these endpoints will be deprecated over the next few releases.

Below is a summary of all the changes made in the release of `petpy 2.0`.

- `petpy` now supports the latest release of Python 3.7
- Support for Python 2.7 is discontinued as Python 2.7 will be officially discontinued in January 2020.

- **The following methods have been added to `petpy` to make it compatible with v2.0 of the Petfinder API.**

  - `animal_types()` is used to getting animal types (or type) available from the Petfinder API. The release of v2.0 of the Petfinder API added several endpoints for accessing animal types in the Petfinder database. This method wraps both Petfinder API endpoints for getting animal types. More information on the animal type endpoints in the Petfinder API can be found in its documentation: - Get Animal Types - Get Single Animal Type
  - `breeds()` is the new method for getting available animal breeds from the Petfinder database. The API endpoint documentation is available on the Petfinder API documentation page. - Get Animal Breeds
  - `animals()` is the method for extracting animal data available on the Petfinder API and deprecates the `pets()` related methods. The method wraps both the `animals` and `animal/{id}` endpoints of the Petfinder API. The documentation for these endpoints can be found in the Petfinder API documentation: - Get Animal - Get Animals
  - `organizations()` is now the method for extracting animal welfare organization data available on Petfinder and deprecates previous `shelter()` related methods and endpoints. The `organizations()` method wraps both the Petfinder API `organizations` and `organizations/{id}` endpoints. The Petfinder API documentation for these two endpoints can be found below: - Get Organizations - Get Organization

- The following methods have been removed as they are no longer valid endpoints with the release of v2.0 of the PetFinder API. - `pet_get_random()` - `shelter_list_by_breed()` - `shelter_get_pets()`
- General refactoring and code clean-up.

4.3.9 Version 1.8.2

- Add `pandas` back as an installation requirement as it is included in top-level imports. `pandas` is still not necessary to use the primary functionality of `petpy`.

4.3.10 Version 1.8.1

- Implement check to make sure `count` parameter is not larger than 1,000 records (per the Petfinder API limits). If `count` exceeds 1,000 a `ValueError` is raised.
- Add check for ensuring the number of total records to return does not exceed 2,000 when paging results with the `pages` and `count` parameters. If the desired amount of records is higher than 2,000, a `ValueError` is raised.
- Remove Python 3.3 support. Although `petpy` should work fine for those still using Python 3.3, testing for 3.3 has been discontinued.

4.3.11 Version 1.8.0

- General refactoring of the `petpy` library to remove unneeded methods from being exposed when importing the library. The best way to import and begin using `petpy` is `from petpy import Petfinder` or, less optimally, `import petpy`, then calling the `Petfinder` class by `petpy.Petfinder`.

4.3.12 Version 1.7.2

- There is now a proper message when the daily API call limit is exceeded. Before the change, methods would return a `JSONDecodeError`.
- The Python 2 to 3 compatibility package `six`, has been added as a requirement for package installation. The `six` library is lightweight and ensures the `petpy` package works properly for Python 2 and 3.
- Numpy is no longer required for installing the package. Numpy's `nan` was initially used to denoted shelters and animals that were not found in the Petfinder database. The value returned when a shelter or animal is not found is now 'na'.

4.3.13 Version 1.7.1

- Fix to the `shelter_get()` method for handling empty responses when no shelters returned for when the parameter `return_df = True`.
- Fix to getting pets available at a shelter with `shelter_get_pets()` when the parameter `return_df = True` when the given shelter does not return any available animals.

4.3.14 Version 1.7.0

- Refactoring of the library to clean up code files.
- Fixed a bug with the `shelter_get_pets()` method that caused an error to be thrown when there is only one pet record and the parameter `return_df = True`.
- Many changes to simplify expressions and internal code within methods.
- The Petfinder class method names and parameters have been renamed to be PEP8 compatible. I apologize as this will cause backward compatibility issues upon updating for anyone using previous versions. The original intention of the naming was to reflect the Petfinder API's method names as closely as possible, but after further consideration and given the relatively short life of the library, I think the change is necessary for the future growth and maturity of the package.
- How the methods interact with the Petfinder API is unchanged. Thus there is no immediate need to update your version of petpy if it will break any current code.
- The Github repo README has been updated with the new API methods.
- Below is a table detailing the changed method names:

Previous Method Name	New Method Name
pet_getRandom()	pet_get_random()
shelter_getPets()	shelter_get_pets()
shelter_listByBreed()	shelter_list_by_breed()

- The following lists the method parameter names that have changed with the release:

Previous Parameter Name	New Parameter Name
petId	pet_id
shelterId	shelter_id

4.3.15 Version 1.6.0

- This release removes pandas as an installation requirement for the package. Although pandas is required to convert the API results into a DataFrame, this is optional and not necessary to the building or use of the package itself.

4.3.16 Version 1.5.995

- Calls that return JSON results when using the `pet_find()` method when `return_df=True` are now adequately handled and an empty pandas DataFrame is returned. This result can happen when searching for a particular breed of animal that is currently not available in the Petfinder database.

4.3.17 Version 1.5.92

- The paged results should now cap at Petfinder's 2,000 search limit consistently.
- The methods `shelter_get()` and `shelters_get()` now handle shelters that have opted-out of having their information available in the Petfinder API.

4.3.18 Version 1.5.91

- Paged results will now reach Petfinder's 2,000 records per search limit. Before, if the next paged result would equal or exceed 2,000 results the call would end, and the results would be returned. For example, if the parameters `pages` is 10 and `count` is 200, 2,000 records will now be returned, whereas previously 1,800 would be returned.

4.3.19 Version 1.5.9

- Paging results that exceed Petfinder's limit of 2,000 records returned per search with `return_df = True` will now correctly exit the loop and return the results as a DataFrame.

4.3.20 Version 1.5.7

- The fix to returning a DataFrame when paging results is now implemented in this release. Apologies for the oversight, the code change was not made before releasing the previous version.
- The contact information returned with a DataFrame when `return_df = True` now has the prefix 'contact.' removed to make the results cleaner.

4.3.21 Version 1.5.6

- Paging results now returns the stated number of pages in the `pages` parameter. Before, `pages + 1` results were returned.
- Returning pandas DataFrames with methods `pet_find()` and `shelter_find()` should no longer throw `ValueError` (duplicate column name was causing an error in concatenating the list of results into a DataFrame).

4.3.22 Version 1.5.5

- `shelter_getPets()` method now returns a complete flattened pandas DataFrame when the parameter `return_df = True`.

4.3.23 Version 1.5.4

- Slight fix to `pet_getRandom()` method. Before, if the method parameter `return_df = True`, but the parameter `output` was not one of 'basic' or 'full', the `return_df` parameter was overridden and set as `False`. Now, if `return_df = True` and `output None`, then `output` is set to 'full' to return the most complete DataFrame.
- Added `records` parameter to `pet_getRandom()` to allow multiple random results to be returned in the same method call. Please note each record returned counts as one call made to the Petfinder API.
- Added API convenience methods `pets_get()` and `shelters_get()` for pulling multiple results given a list or tuple of IDs. These methods are essentially wrappers of the API methods `pet_get()` and `shelter_get()`.
- More code cleanup, formatting, and simplification.

4.3.24 Version 1.5.0

- Add option to convert returned results into a pandas DataFrame.
- Formatting and code cleanup.
- Updated docstrings

4.3.25 Version 1.0.0

First major release.

4.4 Contributors

- ma755 - Fixed several functions that use an `animal` parameter and implementing checks for exceeding the Petfinder API limit.
- ljlevins - Found and fixed an error with the `distance` parameter used in the `organizations` API endpoint.

CHAPTER 5

Introduction

Connecting and using the Petfinder API is as straightforward as initializing the `Petfinder()` class. The following are several examples for extracting data from the Petfinder database and interacting with the Petfinder API.

5.1 Authenticating with the Petfinder API

Authentication to the Petfinder API occurs when the `Petfinder()` class is initialized.

```
import petpy
pf = Petfinder(key=API_key, secret=API_secret)
```

Calls to the API to extract data can now be made!

5.2 Finding Animal Types

```
# All animal types and their relevant data.
all_types = pf.animal_types()

# Returning data for a single animal type
dogs = pf.animal_types('dog')

# Getting multiple animal types at once
cat_dog_rabbit_types = pf.animal_types(['cat', 'dog', 'rabbit'])
```

5.3 Get Breeds of Animal Types

```
cat_breeds = pf.breeds('cat')
dog_breeds = pf.breeds('dog')

# All available breeds or multiple breeds can also be returned.
all_breeds = pf.breeds()
cat_dog_rabbit = pf.breeds(types=['cat', 'dog', 'rabbit'])
```

The *breeds* method can also be set to coerce the returned JSON results into a pandas DataFrame by setting the parameter *return_df = True*.

```
cat_breeds_df = pf.breeds('cat', return_df = True)
all_breeds_df = pf.breeds(return_df = True)
```

5.4 Getting animals on Petfinder

The `animals()` method returns animals based on specified criteria that are listed in the Petfinder database. Specific animals can be searched using the `animal_id` parameter, or a search of the database can be performed by entering the desired search criteria.

```
# Getting first 20 results without any search criteria
animals = pf.animals()

# Extracting data on specific animals with animal_ids

animal_ids = []
for i in animals['animals'][0:3]:
    animal_ids.append(i['id'])

animal_data = pf.animals(animal_id=animal_ids)

# Returning a pandas DataFrame of the first 150 animal results
animals = pf.animals(results_per_page=50, pages=3, return_df=True)
```

5.5 Getting animal welfare organizations in the Petfinder database

Similar to the `animals()` method described above, the `organizations()` method returns data on animal welfare organizations listed in the Petfinder database based on specific criteria, if any. In addition to a general search of animal welfare organizations, specific organizational data can be extracted by supplying the `organizations()` method with organization IDs.

```
# Return the first 1,000 animal welfare organizations as a pandas DataFrame
organizations = pf.organizations(results_per_page=100, pages=10, return_
↪df=True)

# Get organizations in the state of Washington
wa_organizations = pf.organizations(state='WA')
```

CHAPTER 6

Tutorials and Examples

The following are Jupyter Notebooks (launched in Github) that introduce the petpy package and some examples of its usage. The notebooks can also be launched in an interactive environment with binder

- Introduction to petpy
- Download 45,000 Cat Images in 6.5 Minutes with petpy and multiprocessing
  - Please note the following notebook is still based on the legacy version of the Petfinder API and thus are not fully representative of the functionality and methods of the most recent version of `petpy` and the Petfinder API. These are currently being updated to reflect the new version of `petpy`.
- Download Pure Breeds Cat Images with petpy for Deep Neural Network training
  - Provided by contributor ma755

The following are longer usage examples and tutorials that have been posted to external media websites such as Medium.com:

- Analyze Petfinder Adoptable Pet Descriptions with the IBM Watson Tone Analyzer — Part One

Index