uritemplate Documentation

Release 4.1.1

Ian Stapleton Cordasco

Sep 16, 2023

CONTENTS

1 Examples 3

2 API 5

3 Implementation Details 9

Python Module Index 11

Index 13

Release v4.1.1.

CHAPTER

ONE

EXAMPLES

This first example shows how simple the API can be when using for a one-off item in a script or elsewhere.

```
from requests import get
from uritemplate import expand

uri = 'https://api.github.com{/user}'

user = get(expand(uri, user='sigmavirus24')).json()
```

This second example shows how using the class will save you time for template parsing and object creation. Making the template once means the URI is parsed once which decreases the number of `URITemplate` objects created and usage of the `re` module. This means that as soon as the file is parsed, the `User.github_url` and `Repository.github_url` variables are made once and only once. They're then usable in every instance of those classes.

```
from uritemplate import URITemplate

class User(object):
    github_url = URITemplate('https://api.github.com{/user}')
    def __init__(self, name):
        self.uri = self.github_url.expand({'user': name})
        self.name = name

class Repository(object):
    github_url = URITemplate('https://api.github.com{/user}{/repo}')
    def __init__(self, name):
        self.uri = self.github_url.expand(
            dict(zip(['user', 'repo'], name.split('/')))
        )
        self.name = name
```

CHAPTER

TWO

API

uritemplate.api.**expand**(*uri: str*, *var_dict: Optional[Dict[str, Union[Sequence[Union[int, float, complex, str]], Mapping[str, Union[int, float, complex, str]], Tuple[str, Union[int, float, complex, str]], int, float, complex, str]]] = None*, ***kwargs: Union[Sequence[Union[int, float, complex, str]], Mapping[str, Union[int, float, complex, str]], Tuple[str, Union[int, float, complex, str]], int, float, complex, str]*) → str

Expand the template with the given parameters.

Parameters

- **uri** (*str*) – The templated URI to expand
- **var_dict** (*dict*) – Optional dictionary with variables and values
- **kwargs** – Alternative way to pass arguments

Returns

str

Example:

```
expand('https://api.github.com{/end}', {'end': 'users'})
expand('https://api.github.com{/end}', end='gists')
```

Note: Passing values by both parts, may override values in `var_dict`. For example:

```
expand('https://{var}', {'var': 'val1'}, var='val2')
```

`val2` will be used instead of `val1`.

---

uritemplate.api.**partial**(*uri: str*, *var_dict: Optional[Dict[str, Union[Sequence[Union[int, float, complex, str]], Mapping[str, Union[int, float, complex, str]], Tuple[str, Union[int, float, complex, str]], int, float, complex, str]]] = None*, ***kwargs: Union[Sequence[Union[int, float, complex, str]], Mapping[str, Union[int, float, complex, str]], Tuple[str, Union[int, float, complex, str]], int, float, complex, str]*) → *URITemplate*

Partially expand the template with the given parameters.

If all of the parameters for the template are not given, return a partially expanded template.

Parameters

- **var_dict** (*dict*) – Optional dictionary with variables and values
- **kwargs** – Alternative way to pass arguments

Returns
URITemplate

Example:

```
t = URITemplate('https://api.github.com{/end}')
t.partial()  # => URITemplate('https://api.github.com{/end}')
```

uritemplate.api.**variables**(*uri: str*) → OrderedSet

Parse the variables of the template.

This returns all of the variable names in the URI Template.

Returns
Set of variable names

Return type
set

Example:

```
variables('https://api.github.com{/end})
# => {'end'}
variables('https://api.github.com/repos{/username}{/repository}')
# => {'username', 'repository'}
```

class uritemplate.template.**URITemplate**(*uri: str*)

This parses the template and will be used to expand it.

This is the most important object as the center of the API.

Example:

```
from uritemplate import URITemplate
import requests


t = URITemplate(
    'https://api.github.com/users/sigmavirus24/gists{/gist_id}'
)
uri = t.expand(gist_id=123456)
resp = requests.get(uri)
for gist in resp.json():
    print(gist['html_url'])
```

Please note:

```
str(t)
# 'https://api.github.com/users/sigmavirus24/gists{/gistid}'
repr(t)  # is equivalent to
# URITemplate(str(t))
# Where str(t) is interpreted as the URI string.
```

Also, URITemplates are hashable so they can be used as keys in dictionaries.

expand(*var_dict: Optional[Dict[str, Union[Sequence[Union[int, float, complex, str]], Mapping[str, Union[int, float, complex, str]], Tuple[str, Union[int, float, complex, str]], int, float, complex, str]]] = None*, ***kwargs: Union[Sequence[Union[int, float, complex, str]], Mapping[str, Union[int, float, complex, str]], Tuple[str, Union[int, float, complex, str]], int, float, complex, str]*) → str

Expand the template with the given parameters.

Parameters

- **var_dict** (*dict*) – Optional dictionary with variables and values
- **kwargs** – Alternative way to pass arguments

Returns
str

Example:

```
t = URITemplate('https://api.github.com{/end}')
t.expand({'end': 'users'})
t.expand(end='gists')
```

Note: Passing values by both parts, may override values in `var_dict`. For example:

```
expand('https://{var}', {'var': 'val1'}, var='val2')
```

`val2` will be used instead of `val1`.

---

partial(*var_dict: Optional[Dict[str, Union[Sequence[Union[int, float, complex, str]], Mapping[str, Union[int, float, complex, str]], Tuple[str, Union[int, float, complex, str]], int, float, complex, str]]] = None*, ***kwargs: Union[Sequence[Union[int, float, complex, str]], Mapping[str, Union[int, float, complex, str]], Tuple[str, Union[int, float, complex, str]], int, float, complex, str]*) → *URITemplate*

Partially expand the template with the given parameters.

If all of the parameters for the template are not given, return a partially expanded template.

Parameters

- **var_dict** (*dict*) – Optional dictionary with variables and values
- **kwargs** – Alternative way to pass arguments

Returns
URITemplate

Example:

```
t = URITemplate('https://api.github.com{/end}')
t.partial()  # => URITemplate('https://api.github.com{/end}')
```

uri: str

The original URI to be parsed.

variable_names

A set of variable names in the URI.

variables: List[*URIVariable*]

A list of the variables in the URI. They are stored as *URIVariable*s

CHAPTER

THREE

IMPLEMENTATION DETAILS

Classes, their methods, and functions in this section are not part of the API and as such are not meant to be used by users of `uritemplate.py`. These are documented here purely for reference as they are inadvertently exposed via the public API.

For example:

```
t = URITemplate('https://api.github.com/users{/user}')
t.variables
# => [URIVariable(/user)]
```

Users can interact with *URIVariable* objects as they see fit, but their API may change and are not guaranteed to be consistent across versions. Code relying on methods defined on *URIVariable* and other classes, methods, and functions in this section may break in future releases.

class uritemplate.variable.**URIVariable**(*var: str*)

This object validates everything inside the URITemplate object.

It validates template expansions and will truncate length as decided by the template.

Please note that just like the `URITemplate`, this object's `__str__` and `__repr__` methods do not return the same information. Calling `str(var)` will return the original variable.

This object does the majority of the heavy lifting. The `URITemplate` object finds the variables in the URI and then creates `URIVariable` objects. Expansions of the URI are handled by each `URIVariable` object. `URIVariable.expand()` returns a dictionary of the original variable and the expanded value. Check that method's documentation for more information.

expand(*var_dict: Optional[Dict[str, Union[Sequence[Union[int, float, complex, str]], Mapping[str, Union[int, float, complex, str]], Tuple[str, Union[int, float, complex, str]], int, float, complex, str]]] = None*) → Mapping[str, str]

Expand the variable in question.

Using `var_dict` and the previously parsed defaults, expand this variable and subvariables.

Parameters
var_dict (*dict*) – dictionary of key-value pairs to be used during expansion

Returns
dict(variable=value)

Examples:

```
# (1)
v = URIVariable('/var')
expansion = v.expand({'var': 'value'})
```

(continues on next page)

(continued from previous page)

```
print(expansion)
# => {'/var': '/value'}

# (2)
v = URIVariable('?var,hello,x,y')
expansion = v.expand({'var': 'value', 'hello': 'Hello World!',
                      'x': '1024', 'y': '768'})
print(expansion)
# => {'?var,hello,x,y':
#     '?var=value&hello=Hello%20World%21&x=1024&y=768'}
```

PYTHON MODULE INDEX

u

uritemplate, 5

INDEX

E

expand() (*in module uritemplate.api*), 5
expand() (*uritemplate.template.URITemplate method*), 6
expand() (*uritemplate.variable.URIVariable method*), 9

M

module
    uritemplate, 5

P

partial() (*in module uritemplate.api*), 5
partial() (*uritemplate.template.URITemplate method*), 7

U

uri (*uritemplate.template.URITemplate attribute*), 7
uritemplate
    module, 5
URITemplate (*class in uritemplate.template*), 6
URIVariable (*class in uritemplate.variable*), 9

V

variable_names (*uritemplate.template.URITemplate attribute*), 7
variables (*uritemplate.template.URITemplate attribute*), 7
variables() (*in module uritemplate.api*), 6