Python API Cheatsheet

With the DNSimple library you can easily interact our powerful API to administer domain names, configure DNS records, provision and install SSL certificates, and more.
Getting Started

1. Install the Python library

```bash
pip install dnsimple
```

2. Authenticate

Obtain your API access token: [https://support.dnsimple.com/articles/api-access-token/](https://support.dnsimple.com/articles/api-access-token/)

```python
client = Client(sandbox=True, access_token='AUTH_TOKEN')
```

3. Check Authorization

If you want to know which account is associated with the current access token, you can use `identity`. The account ID is required for the majority of API operations.

```python
response = client.identity.whoami()
account = response.data.account
let account_id = accound.id;

print(f'{account.id} (your account ID)')

# => 1234 (your account ID)
Managing Domains

Check Domain Availability

Check if a domain is available for registration.

```python
response = client.registrar.check_domain(account_id, 'foo.com')
domain_check = response.data

print(f'Domain: {domain_check.domain}
Available: {domain_check.available}
Premium: {domain_check.premium}')
# => Domain: foo.com
# Available: true
# Premium: false
```

Register A Domain

1. To register a domain, you need to specify a registrant_id. This can be fetched via the Contacts API.

```python
contacts = client.contacts.list_contacts(account_id).data
first_contact = contacts[0]

print(f'{first_contact.id}')
# => 123
```

2. You can register the domain with this information.

```python
domain = client.registrar.register_domain(account_id, 'foo.com',
                                         DomainRegistrationRequest(first_contact.id)).data

print(f'State: {domain.state}
Auto Renew: {domain.auto_renew}
Whois Privacy: {domain.whois_privacy}
Registrant: {domain.registrant_id}')
# => State: registered
# Auto Renew: false
# Whois Privacy: false
# Registrant: 123
```
Create a DNS record

Create a DNS A record to map an IP address to a domain.

```python
input = ZoneRecordInput('www', 'A', '127.0.0.1')
record = client.zones.create_record(account_id, 'foo.com', input).data

print(f'
ID: {record.id}
Zone: {record.zone_id}
Name: {record.name}
Type: {record.type}
Content: {record.content}
')
```

# =>ID: 123
# Zone: foo.com
# Name: www
# Type: A
# Content: 137.0.0.1
Update a DNS record

Update a previously created DNS record.

```python
input = ZoneRecordUpdateInput(ttl='60')
updated_record = client.zones.update_record(account_id, 'foo.com', record.id, input).data
print(f'ID: {updated_record.id}\nUpdated TTL: {updated_record.ttl}"
# =>ID: 123
# Updated TTL: 60
```
SSL Certificates

Order an SSL Certificate with Let's Encrypt

Creates the purchase order. Use the ID to issue the certificate.

```python
input = LetsencryptCertificateInput(
    auto_renew=false, name='test-certificate')

certificate = client.certificates.purchase_letsencrypt_certificate(
    account_id, 'foo.com', input)

print(f'ID: {certificate.id}
State: {certificate.state}')

# => ID: 123
# State: new
```
Issue an Let's Encrypt Certificate

Issues the pending order. This process is async. A successful response means that the response is queued.

```python
issued = client.certificates.issue_letsencrypt_certificate(account_id, 'foo.com', certificate.id).data

print(f'State: {issued.state}')

# =>State: requesting
```
Install the certificate

Download the certificate.

certificate = client.certificates.download_certificate(account_id, 'foo.com', certificate.id).data

defile = open('www_foo_com.pem', 'w')
defile.write(certificate.server)
defile.write('
')
defile.write('
'.join(certificate.chain))
defile.close

Download the certificate's private key.

key = client.certificates.get_certificate_private_key(account_id, 'foo.com', certificate.id).data

defile = open('www_foo_com.key')
defile.write(key.private_key)
defile.close

dnsimple

Get Help From Developers

We provide worry-free DNS services to simplify your life.

Try us free for 30 days