dnsimple

Ruby API Cheatsheet

With the DNSimple gem 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 Ruby gem

```
gem install dnsimple
```

2. Authenticate

Obtain your API access token: https://support.dnsimple.com/articles/api-access-token/

```
require 'dnsimple'
client = Dnsimple::Client.new(access_token: "your-api-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.

```
identity = client.identity.whoami.data
puts identity.account.id
=> 1234 (your account ID)
```

Managing Domains

Check Domain Availability

Check if a domain is available for registration.

```
check = client.registrar.check_domain(account_id, "foo.com")
puts check.data.available
=> true
```

Register A Domain

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

```
contacts = client.contacts.contacts(account_id)
puts contacts.data.first.id
=> 123
```

2. You can register the domain with this information.

```
registration = client.registrar.register_domain(
    account_id,
    "foo.com",
    registrant_id: contact_id
).data
puts registration
=> #<Dnsimple::Struct::DomainRegistration
      @state="registered",
      @auto_renew=false,
      @whois_privacy=false,
      @period=1,
      @registrant_id=123>
```

DNS

Create a DNS record

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

```
record = client.zones.create_zone_record(
    account_id,
    "foo.com",
    name: "www",
    type: "A",
    content: "127.0.0.1"
).data
puts record.id
=> 123
```

Update a DNS record

Update a previously created DNS record.

```
updated = client.zones.update_zone_record(
    account_id,
    "foo.com",
    record.id,
    ttl: 60
).data
puts updated.ttl
=> 60
```

SSL Certificates

Order an SSL Certificate with Let's Encrypt

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

```
attributes = {}
cert = client.certificates.purchase_letsencrypt_certificate(
    account_id,
    "foo.com",
    attributes
).data
puts cert.id
=> 123
puts cert.common_name
=> "www.foo.com"
puts cert.authority_identifier
=> "letsencrypt"
```

Issue an Let's Encrypt Certificate

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

```
client.certificates.issue_letsencrypt_certificate(
    account_id,
    "foo.com",
    cert.id
).data
puts cert.state
=> "requesting"
```

Install the certificate

Download the certificate.

```
cert = client.certificates.download_certificate(
    account_id,
    "foo.com",
    certificate.id
).data
File.open("www_foo_com.pem", "wb") do |file|
  file.write(cert.server)
  file.write("\n")
  file.write(cert.chain.join("\n"))
end
```

Download the certficate's private key.

```
key = client.certificates.certificate_private_key(
    account_id,
    "foo.com",
    certificate.id
).data
File.open("www_foo_com.key", "wb") do |file|
  file.write(key.private_key)
end
```