dnsimple

Node.js 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.

⚠️ Examples shown are in ECMAScript 6, but our API is compatible with ES5.

Getting Started

1. Install via package manager

`yarn add dnsimple` or `npm install dnsimple`

2. Authenticate

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

```
const client = require("dnsimple")({ accessToken: "abc123" });
```

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.

```
account = await client.identity.whoami();
console.log(account.data);

// { account: { id: 1010, email: 'your@email.com', ... }
```

Managing Domains

Check Domain Availability

Check if a domain is available for registration.

```
response = await client.registrar.checkDomain(accountId, "dnsimple.com");
console.log(response.data);

// { domain: 'dnsimple.com', available: false, 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.

```
response = await client.contacts.listContacts(accountId);
console.log(response.data);

// [{ id: 12345, first_name: 'Jane', last_name: 'Test', email: 'jane@test.c
```

2. You can register the domain with this information.

```
const attributes = {
    registrant_id: 12345,
    whois_privacy: true,
    auto_renew: true
}
response = await client.registrar.registerDomain(
    accountId,
    "example.com",
    attributes);
console.log(response.data);

// { id: 34567,
//  domain_id: 67890,
//  registrant_id: 12345,
//  period: 1,
//  state: 'new',
//  auto_renew: true,
//  whois_privacy: true, ... }
```

DNS

Create a DNS record

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

```
const attributes = {
    name: "",
    type: "A",
    content: "1.2.3.4"
}
response = await client.zones.createZoneRecord(
  accountId,
  "foo.com",
  attributes);
  console.log(response.data);

// { id: 12345,
//  zone_id: 'foo.com',
//  name: 'test',
//  content: '1.2.3.4',
//  ttl: 3600,
//  type: 'A', ...}
```

Update a DNS record

Update a previously created DNS record.

```
const recordId = 12345;
const attributes = { ttl: 60 }
response = await client.zones.updateZoneRecord(
  accountId,
  "foo.com",
  recordId,
  attributes);
console.log(response.data);

// { id: 12345,
//  zone_id: 'foo.com',
//  name: 'test',
//  content: '1.2.3.4',
//  ttl: 60,
//  type: 'A', ... }
```

SSL Certificates

Order an SSL Certificate with Let's Encrypt

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

```
response = await client.certificates.purchaseLetsencryptCertificate(
  accountId,
  "foo.com");
console.log(response.data);

// { id: 12345,
//  common_name: 'www',
// }
```

Issue an Let's Encrypt Certificate

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

```
const certificatePurchaseId = 1234;
client.certificates.issueLetsencryptCertificate(
  accountId,
  "foo.com",
  certificatePurchaseId);
console.log(response.data);

// { id: 12345,
//  common_name: 'www',
//  state: 'requesting',
// }
```

Install the certificate

Download the certificate.

```
const certificateId = 12345;
fs = require("fs");
response = await client.certificates.downloadCertificate(
  accountId,
  "foo.com",
  certificateId);

fs.writeFile(
    "foo_com.pem",
    `${response.data.server}\n${response.data.chain}`);
```

Download the certficate's private key.

```
const certificateId = 27059;
response = await client.certificates.getCertificatePrivateKey(
  accountId,
  "foo.com",
  certificateId);

fs.writeFile("foo_com.key", response.data.private_key);
```

dnsimple

Get Help From Developers

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

Try us free for 30 days