Elixir API Cheatsheet

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

1. Add the DNSimple API client as a dependency to your `mix.exs` file.

```elixir
defp deps do
  [{:dnsimple, "~> 1.0.0"}]
end
```

Install the newly added dependency:

```
> mix deps.get
Running dependency resolution ...
```

2. Authenticate

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

```elixir
client = %Dnsimple.Client{access_token: "TOKEN",
                          base_url: "https://api.sandbox.dnsimple.com/"}
```

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.

```elixir
{:ok, response} = client.Identity.whoami(client)
account_id = response.data.account.id
IO.puts(account_id)
=> 1234  (your account ID)
Managing Domains

Check Domain Availability

Check if a domain is available for registration.

```elixir
{:ok, check} = client.Registrar.check_domain(client, account_id, "foo.com")
IO.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.

```elixir
attributes = %{
  registrant_id: 123,
  auto_renew: false, 
  whois_privacy: false
}
```

2. You can register the domain with this information.

```elixir
{:ok, response} = client.Registrar.register_domain(
  account_id, 
  "foo.com", attributes)
registration = response.data
IO.puts 
  "State: #{registration.state}\n  Auto Renew: #{registration.auto_renew}\n  Whois Privacy: #{registration.whois_privacy}\n  Registrant: #{registration.registrant_id}"

=>
  State: registered
  Auto Renew: false
  Whois Privacy: false
  Registrant: 123
```
DNS

Create a DNS record

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

```elixir
attributes = %{type: "A", name: "www", content: "127.0.0.1"}
{:ok, response} = client.Zones.create_zone_record(account_id, "foo.com", attributes)
record = response.data
IO.puts record.id
=> 123
```

Update a DNS record

Update a previously created DNS record.

```elixir
attributes = %{ttl: 60}
{:ok, response} = client.Zones.update_zone_record(account_id, "foo.com", record_id = record.id, attributes)
updated = response.data
IO.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.

```elixir
{:ok, response} = client.Certificates.purchase_letsencrypt_certificate(account_id, "foo.com")
cert = response.data
IO.puts "ID: #{cert.id}\nState: #{cert.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.

```elixir
{:ok, response} = client.Certificates.issue_letsencrypt_certificate(account_id, "foo.com", _certificate_id = cert.id)
cert = response.data
IO.puts "State: #{cert.state}"
=>State: requesting
```
Install the certificate

Download the certificate.

```elixir
{:ok, response} = client.certificates.download_certificate(
  account_id, "foo.com", _certificate_id = certificate.id
)
cert = response.data

{:ok, file} = File.open("www_foo_com.pem", [:write])
IO.binwrite(file, cert.server)
IO.binwrite(file, Enum.join(cert.chain, "\n"))
File.close(file)
```

Download the certificate's private key.

```elixir
{:ok, response} = client.certificates.certificate_private_key(
  account_id, "foo.com", _certificate_id = certificate.id
)
key = response.data

{:ok, file} = File.open("www_foo_com.key", [:write])
IO.binwrite(file, key.private_key)
File.close(file)
```

dnsimple

Get Help From Developers

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

Try us free for 30 days