

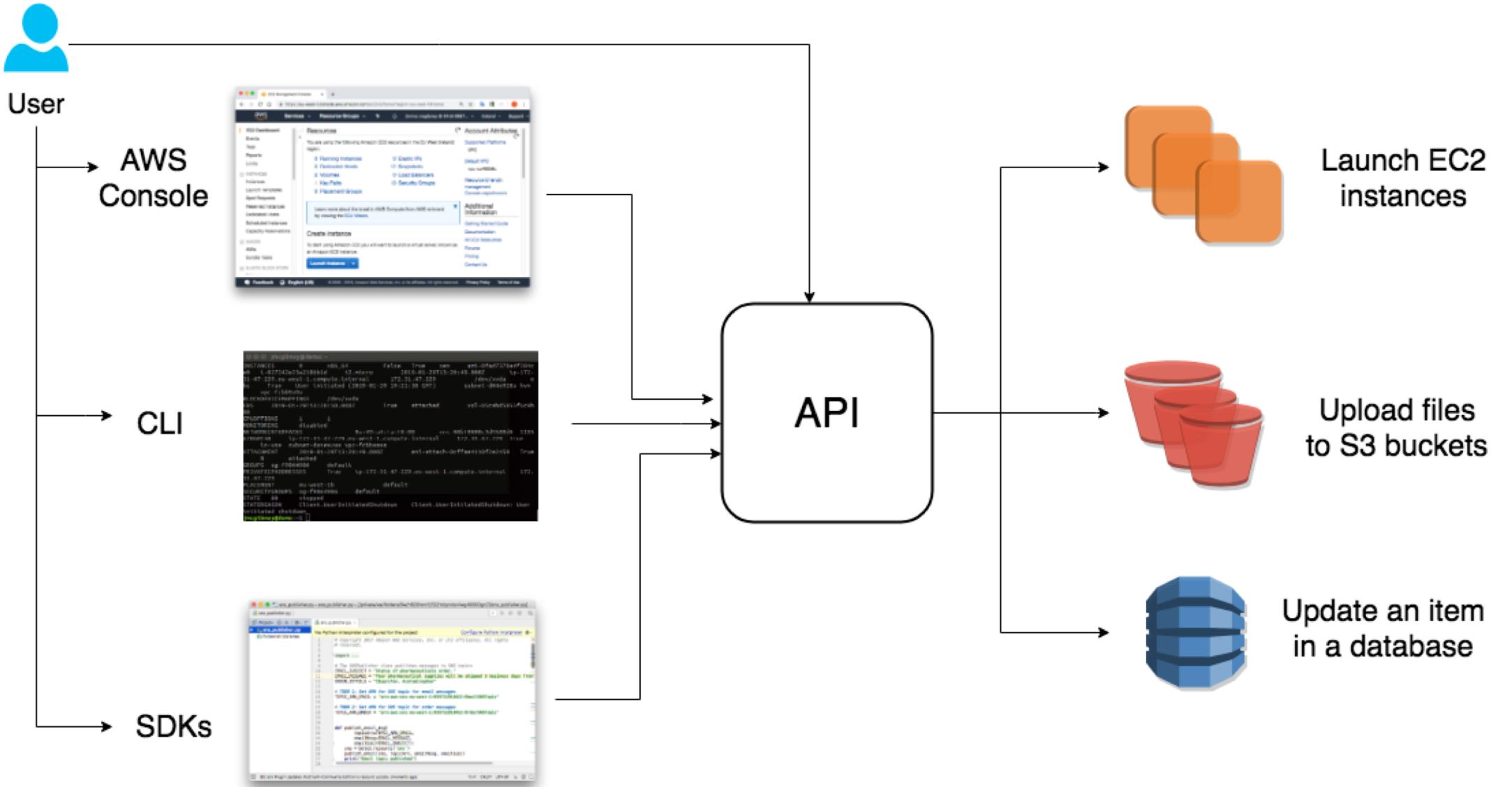
Developer Operations

Developing on AWS:
Overview

Four ways to use AWS

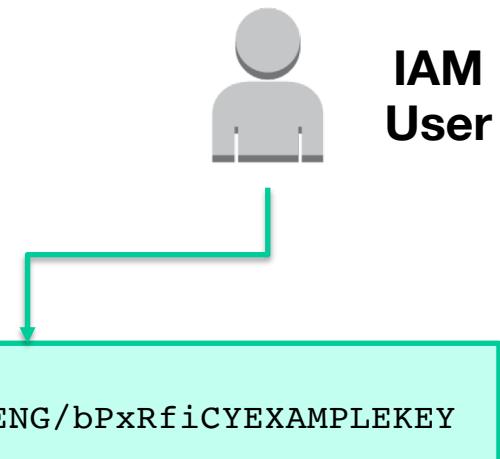
- With AWS, all services are managed through an Application Programming Interface (API).
- A user can access this API in any of 4 ways:
 - Management Console
 - Command Line Interface (CLI)
 - Software Development Kit (SDK)
 - SDKs available for many languages (e.g. Python, Java, .NET)
 - Designed to simplify things for developers (compared with direct API calls)
 - Direct REST-like API calls
 - Using HTTP/HTTPS

Four ways to use AWS



Authentication

- Management Console based on Username and Password (+ optional MFA)
- All other methods based on API credentials
 - Access Key ID & Secret Access Key



AWS CLI

```
jmcgibney@demo:~$ aws ec2 describe-instances
RESERVATIONS 118567060150 r-01d71c3f18124f689
INSTANCES 0 x86_64 False True xen ami-0bce5ae5a9ce
6d623 i-00dc204260426a28e t2.micro jrh 2018-12-12T18:07:22.000Z
          ip-172-31-22-102.eu-west-1.compute.internal 172.31.22.102
/dev/xvda ebs True User initiated (2018-12-12 19:04:17 GMT)
subnet-123d505a hvm vpc-fcbb9a9a
BLOCKDEVICEMAPPINGS /dev/xvda
```

AWS SDK & API



Interoperability

- The various access modes can be used interchangeably
 - e.g. you could create an instance using Python/boto3, start it using the CLI and stop it using the management console
- They all use the API underneath
- Note though that changes made using API may not appear immediately in the console (most pages have a 'refresh' button)

Command Line Interface (CLI) format

Service
(command)

Operation
(sub-command)

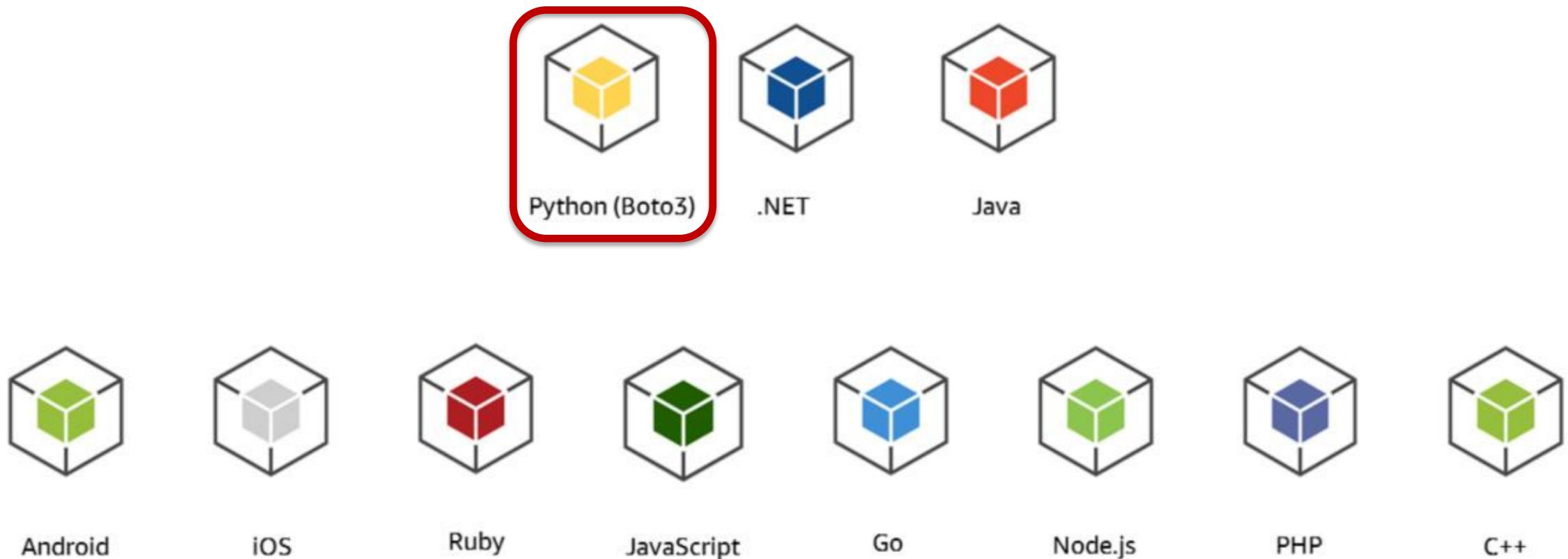
Parameters

```
$ aws ec2 stop-instances --instance-id i-1234567890abcdef0
```

```
$ aws ec2 run-instances --cli-input-json file://./lcf/webserver.json
```

```
$ aws help
$ aws ec2 help
$ aws ec2 describe-instances help
```

Software Development Kit (SDK) libraries for several languages



- We are using the Boto3 SDK for Python in this module
- This is a library of classes and functions that we can use in our Python programs

Connecting to a Service

- Can use Service Client API vs Resource API
- Resource API newer and usually recommended approach

Client API vs Resource API

Service Client API

- Focused on **services**
- More low-level
- What operations can be carried out with the service (e.g. EC2 or S3)
- Has objects for request and result data

```
ec2client = boto3.client('ec2')
```

Client API vs Resource API

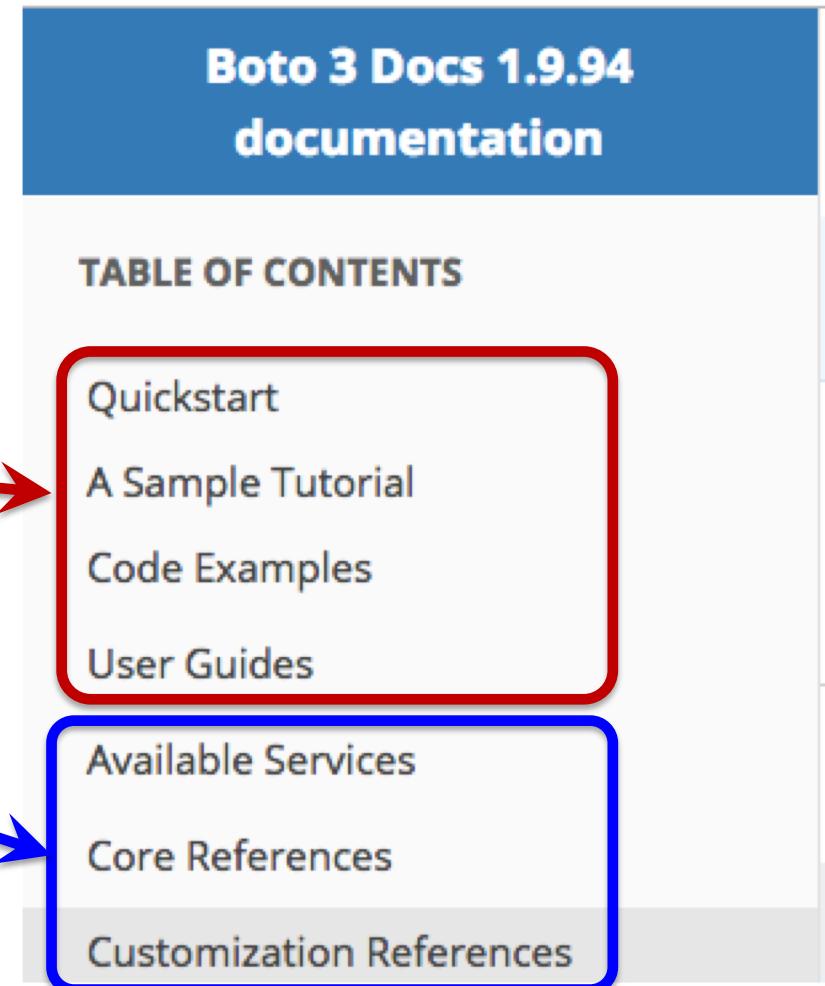
Resource API

- Focused on **resources**
- More high-level, more object oriented
- Has one class per conceptual resource
- Resource can be a service (e.g. EC2, S3) or an individual resource (e.g. an instance, a security group, an S3 bucket)

```
ec2resource = boto3.resource('ec2')
```

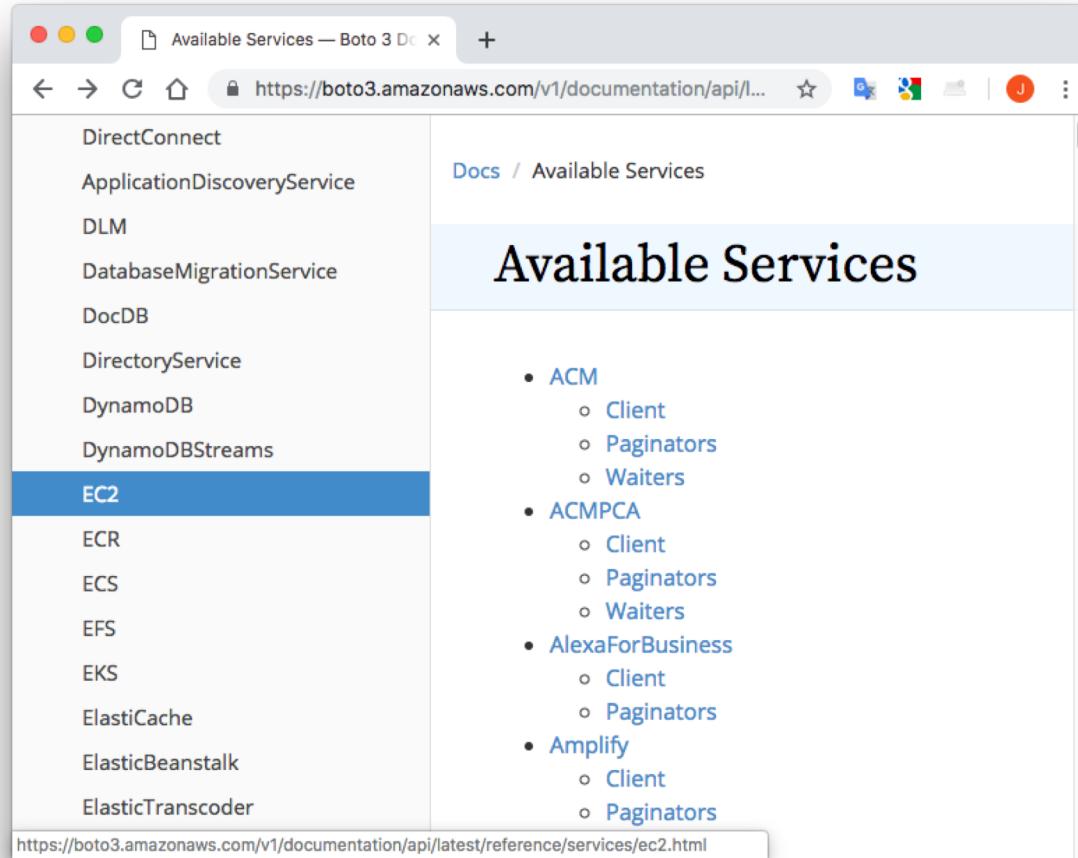
Navigating the Boto3 documentation

- Documentation home:
<https://boto3.readthedocs.io>
- Mix of:
 - User guide / tutorial / examples
 - Full references

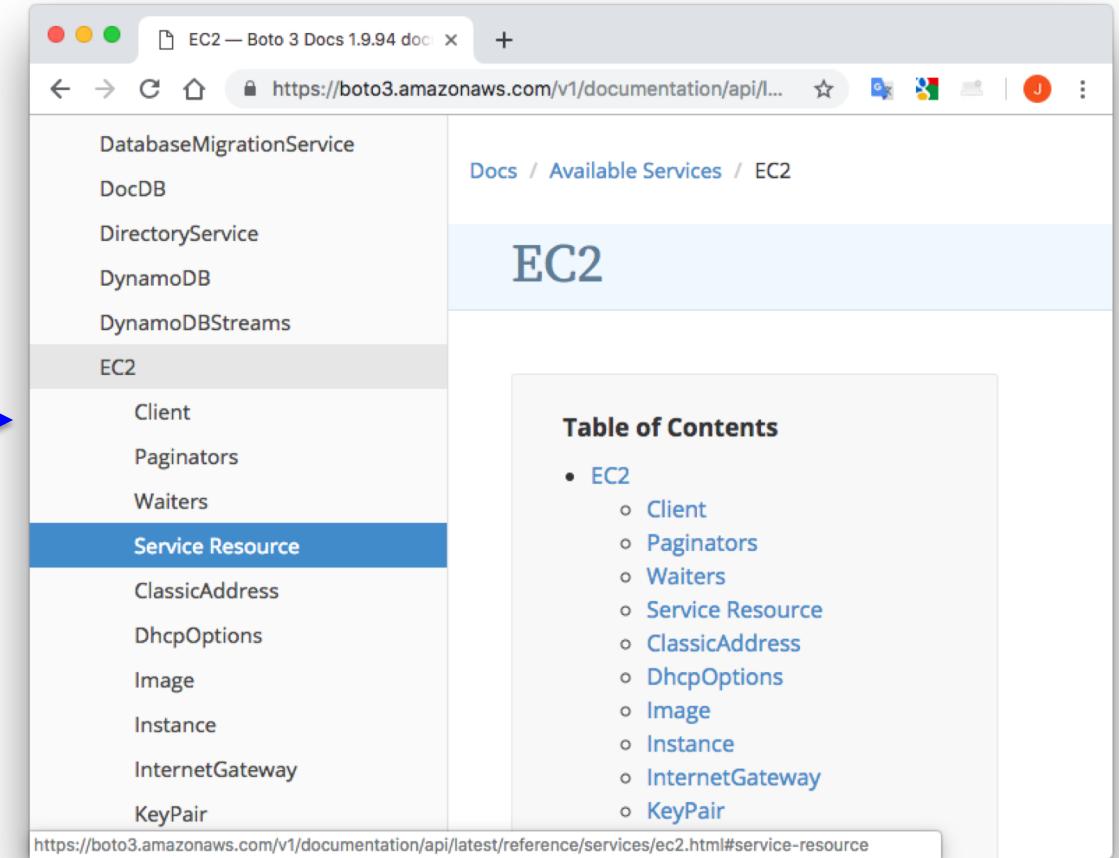


Navigating the Boto3 documentation

- **Available Services** is most useful reference section
 - Select the service you want; e.g. EC2, S3



The screenshot shows a web browser window titled "Available Services — Boto 3 Doc". The URL is <https://boto3.amazonaws.com/v1/documentation/api/latest/reference/services/index.html>. On the left, there is a sidebar with a list of services: DirectConnect, ApplicationDiscoveryService, DLM, DatabaseMigrationService, DocDB, DirectoryService, DynamoDB, DynamoDBStreams, EC2 (which is highlighted with a blue background), ECR, ECS, EFS, EKS, ElastiCache, ElasticBeanstalk, and ElasticTranscoder. The main content area is titled "Available Services" and lists several service names with their corresponding Client, Paginators, and Waiters.



The screenshot shows a web browser window titled "EC2 — Boto 3 Docs 1.9.94 doc". The URL is <https://boto3.amazonaws.com/v1/documentation/api/latest/reference/services/ec2.html#service-resource>. The sidebar on the left shows the EC2 service selected. The main content area is titled "EC2" and contains a "Table of Contents" with items such as Client, Paginators, Waiters, Service Resource (which is highlighted with a blue background), ClassicAddress, DhcpOptions, Image, Instance, InternetGateway, and KeyPair.

Navigating the Boto3 documentation

- And then the specific resource API (or client API)

The screenshot shows a web browser window with the title "EC2 — Boto 3 Docs 1.9.94 doc". The URL in the address bar is <https://boto3.amazonaws.com/v1/documentation/api/latest/r...>. The page content is titled "Service Resource" and describes the `EC2.ServiceResource` class. It states that it represents Amazon Elastic Compute Cloud (EC2). Below the description, there is a code snippet demonstrating how to import the module and create a resource object:

```
import boto3

ec2 = boto3.resource('ec2')
```

Further down, it lists the available actions for the resource:

- `create_dhcp_options()`
- `create_instances()`
- `create_internet_gateway()`
- `create_key_pair()`
- `create_network_acl()`

The left sidebar of the browser shows a navigation menu for the EC2 service, including options like "Client", "Paginators", "Waiters", "Service Resource", and "Instance".

Navigating the Boto3 documentation

- From which you can look at the detail of a specific method, for example

The screenshot shows a web browser window with the title "EC2 — Boto 3 Docs 1.9.94 doc". The URL in the address bar is <https://boto3.amazonaws.com/v1/documentation/api/latest/r...>. The page content is titled "Service Resource" and describes the `EC2.ServiceResource` class. It states that it represents Amazon Elastic Compute Cloud (EC2). Below the description, there is a code snippet:

```
import boto3

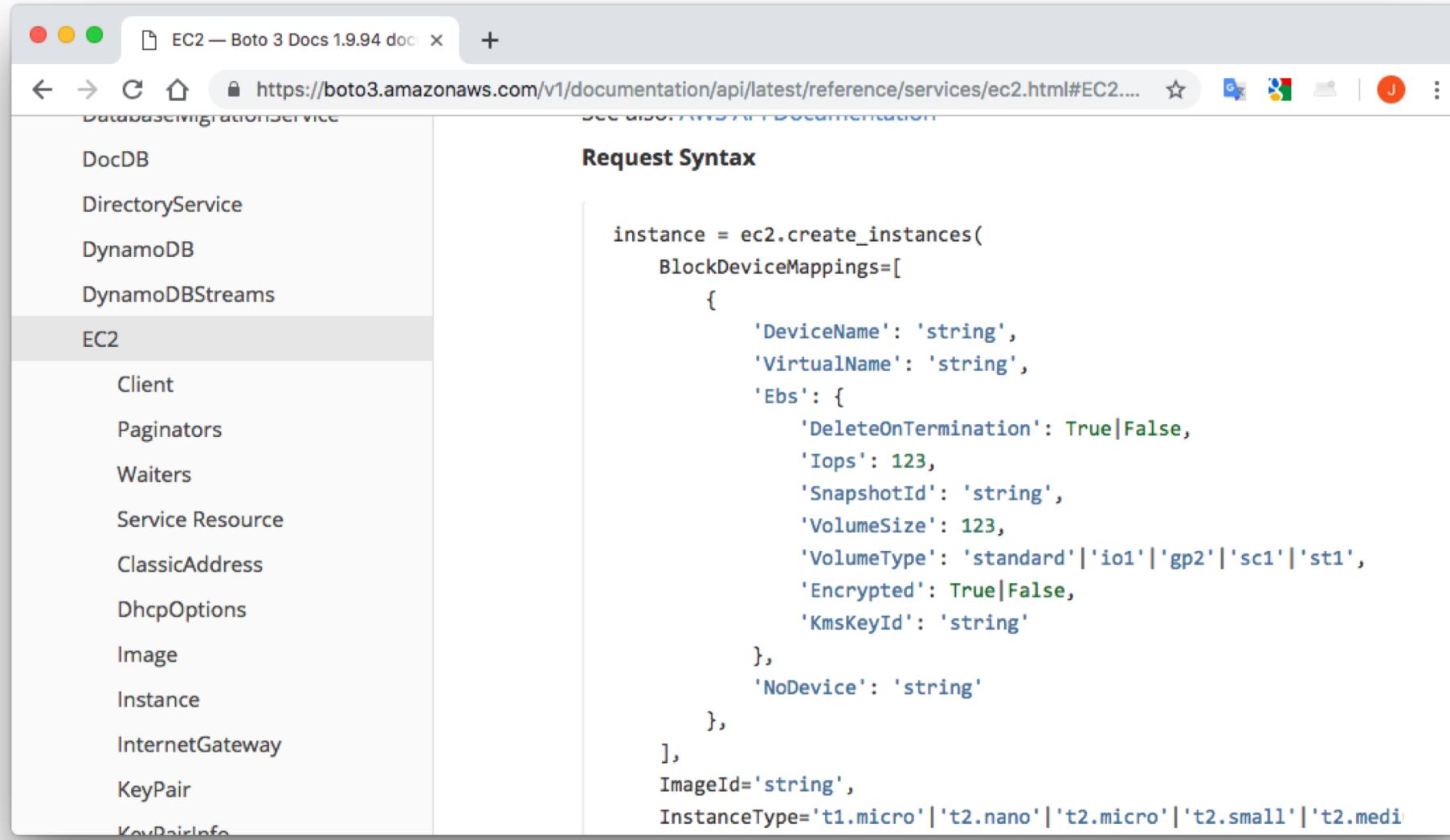
ec2 = boto3.resource('ec2')
```

Underneath the code, it says "These are the resource's available actions:" followed by a bulleted list of methods. One of the methods, `create_instances()`, is highlighted with a red rectangular box.

- `create_dhcp_options()`
- **`create_instances()`**
- `create_internet_gateway()`
- `create_key_pair()`
- `create_network_acl()`

Navigating the Boto3 documentation

- From which you can look at the detail of a specific method, for example



The screenshot shows a web browser window with the title "EC2 — Boto 3 Docs 1.9.94 doc". The URL in the address bar is <https://boto3.amazonaws.com/v1/documentation/api/latest/reference/services/ec2.html#EC2-CreateInstances>. The page content is the "Request Syntax" for the `create_instances` method of the EC2 service. The sidebar on the left lists various services and their components, with "EC2" selected. The main content area contains the following Python code:

```
instance = ec2.create_instances(  
    BlockDeviceMappings=[  
        {  
            'DeviceName': 'string',  
            'VirtualName': 'string',  
            'Ebs': {  
                'DeleteOnTermination': True|False,  
                'Iops': 123,  
                'SnapshotId': 'string',  
                'VolumeSize': 123,  
                'VolumeType': 'standard'|'io1'|'gp2'|'sc1'|'st1',  
                'Encrypted': True|False,  
                'KmsKeyId': 'string'  
            },  
            'NoDevice': 'string'  
        },  
    ],  
    ImageId='string',  
    InstanceType='t1.micro'|'t2.nano'|'t2.micro'|'t2.small'|'t2.medi
```