AWS CDK

Infrastructure **is** Code

Michael Fedell | s.michaelfedell@gmail.com | Nov 12 - 2020

Need to provision cloud resources?

3 common options

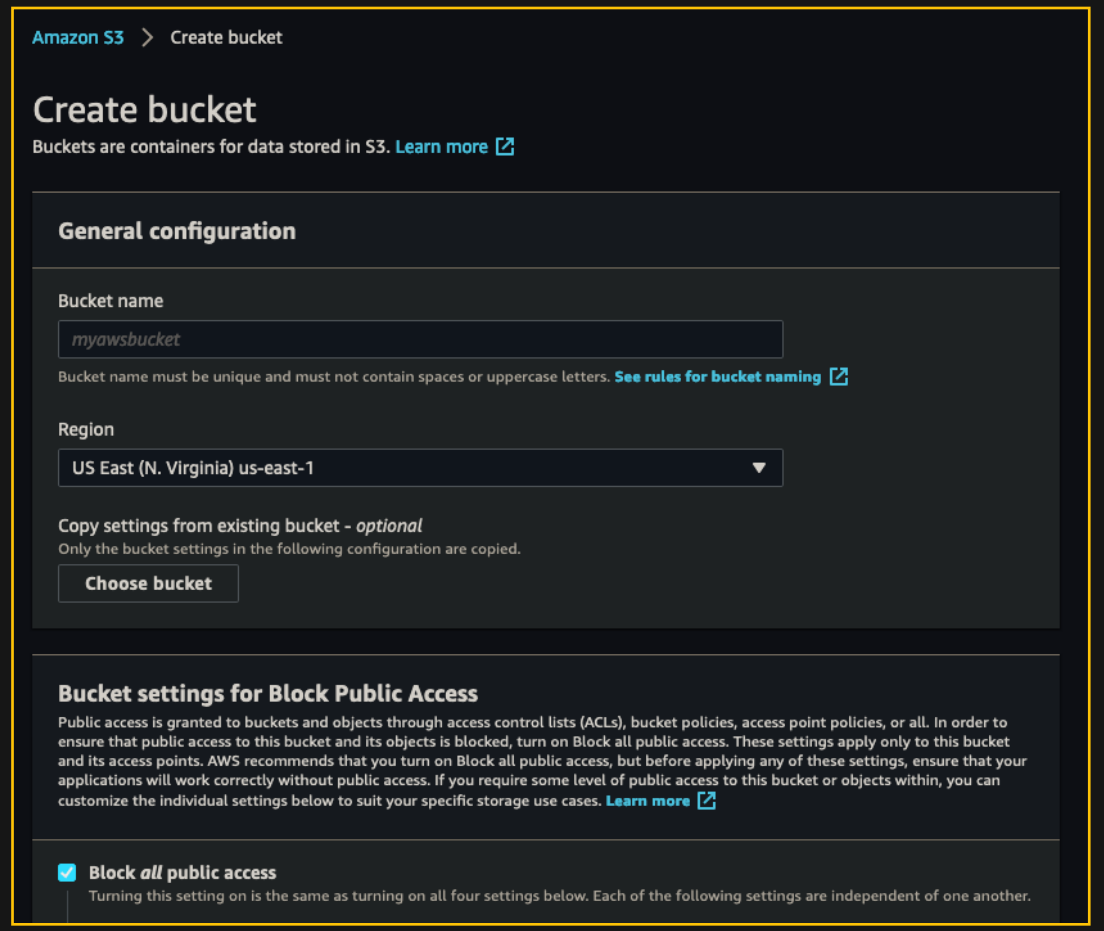
- AWS Console
- AWS SDK's
- Cloudformation Templates

AWS Console

"easy" way to manage applications*

+ Beginner Friendly
+ Great for learning new services
+ Some resources created by default

– Difficult to extend/repeat
– Resources are orphaned
– Lengthy for complex apps*
– Some resources created by default



AWS SDK's

sadistic way to manage applications

+ Can be automated
+ Flexible control
+ Language support

– Individual resource actions
– Must be verbose/explicit
– Still must rely on much documentation

```python
import logging
import boto3
from botocore.exceptions import ClientError


def create_bucket(bucket_name, region=None):
    """Create an S3 bucket in a specified region

    If a region is not specified, the bucket is created in the S3 default
    region (us-east-1).

    :param bucket_name: Bucket to create
    :param region: String region to create bucket in, e.g., 'us-west-2'
    :return: True if bucket created, else False
    """

    # Create bucket
    try:
        if region is None:
            s3_client = boto3.client('s3')
            s3_client.create_bucket(Bucket=bucket_name)
        else:
            s3_client = boto3.client('s3', region_name=region)
            location = {'LocationConstraint': region}
            s3_client.create_bucket(Bucket=bucket_name,
                                    CreateBucketConfiguration=location)
    except ClientError as e:
        logging.error(e)
        return False
    return True
```

Cloudformation Templates

preferred way to manage applications

+ Management, Version Control
+ Collaboration, Sharing
+ "Stack" of related resources

– Can be cumbersome
– Requires extensive documentation reference
– Can be difficult to fully parameterize or extend

```yaml
AWSTemplateFormatVersion: 2010-09-09
Resources:
  S3Bucket:
    Type: AWS::S3::Bucket
    Properties:
      AccessControl: PublicRead
      WebsiteConfiguration:
        IndexDocument: index.html
        ErrorDocument: error.html
    DeletionPolicy: Retain
  BucketPolicy:
    Type: AWS::S3::BucketPolicy
    Properties:
      PolicyDocument:
        Id: MyPolicy
        Version: 2012-10-17
        Statement:
          - Sid: PublicReadForGetBucketObjects
            Effect: Allow
            Principal: '*'
            Action: 's3:GetObject'
            Resource: !Join
              - ''
              - - 'arn:aws:s3:::'
                - !Ref S3Bucket
                - /*
      Bucket: !Ref S3Bucket
```

INTRODUCING CDK

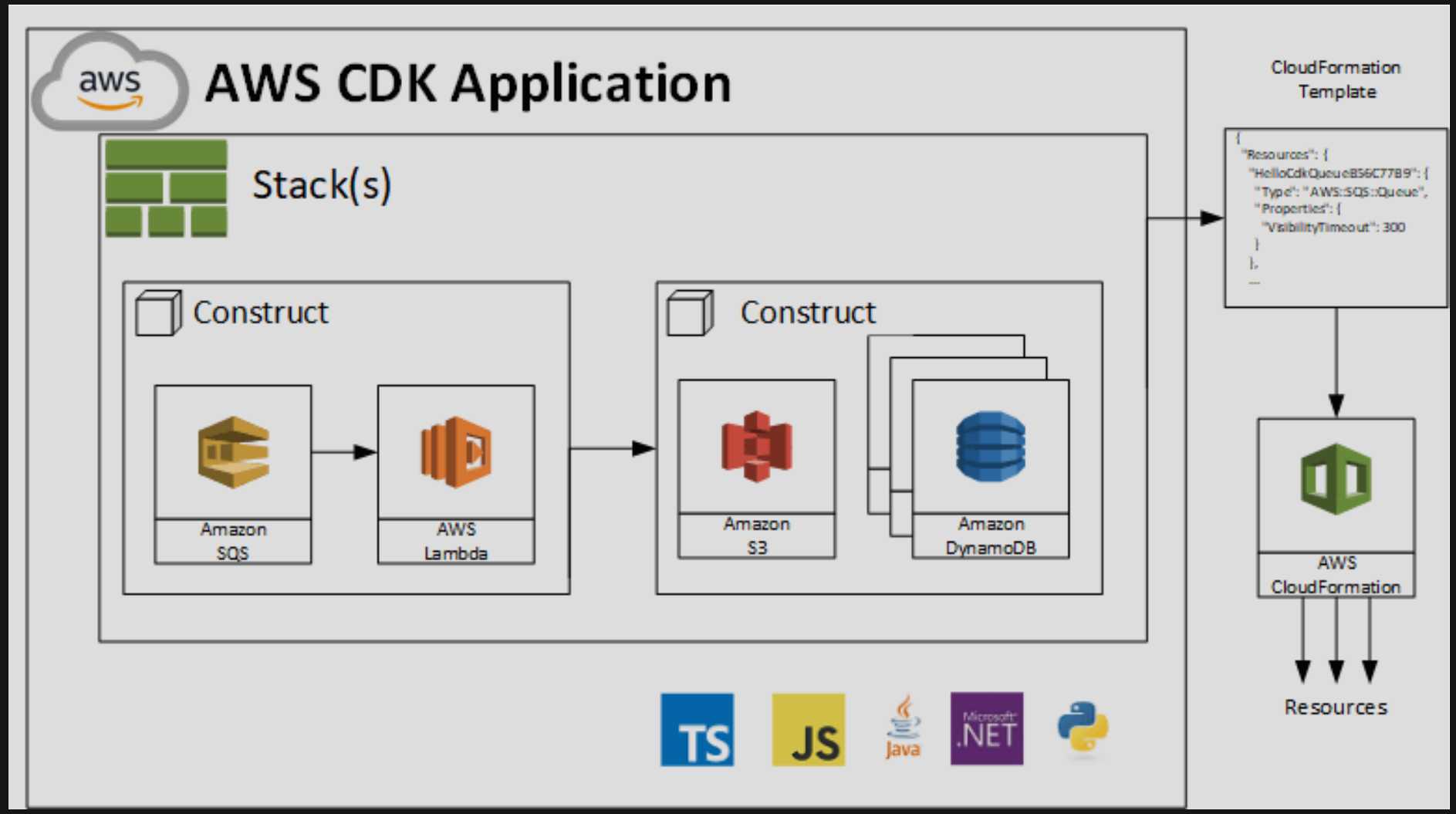
An easy way to define, deploy, and manage complex applications

Great for both application developers and architects

The Basics

- CDK is:
  - A Library (that includes an extensive API for defining resources)
  - A “compiler” (that generates cloudformation from code)
  - A Command Line Tool (that deploys and manages resources)
- CDK lets you define resources using:
  - Typescript (native)
  - Javascript, Python, Java, C# (interpreted to TS via JSII layer)
- A CDK Project consists of:
  - resource[s] > [constructs] > stack[s] > app
- Loosely analogous to moving from writing HTML to React

Visually:


AWS CDK Application
Stack(s)
Construct
Amazon SQS
AWS Lambda
Construct
Amazon S3
Amazon DynamoDB
CloudFormation Template
{
"Resources": {
"HelloCdkQueueB56C77B9": {
"Type": "AWS::SQS::Queue",
"Properties": {
"VisibilityTimeout": 300
}
},
...
AWS CloudFormation
Resources
TS
JS
Java
.NET

Core Concepts

- Resource: Native AWS Resources such as buckets, queues, EC2 instances, etc. Represented in CDK various ways.
- Construct: CDK Building Blocks; can represent single resource or some atomic collection of them
  - L1: CfnResource – direct mapping to Cloudformation spec; full support
  - L2: Resource – intent-based API designed by CDK; extensive, growing support
  - L3: Pattern – high-level API comprising many, configured resources; available for selected common use cases
- Stack: Made up of several constructs; maps to Cloudformation Stack
- Application: Can compose multiple stacks together as single application for easy deployment and management

https://docs.aws.amazon.com/cdk/latest/guide/core_concepts.html

Some Big Advantages

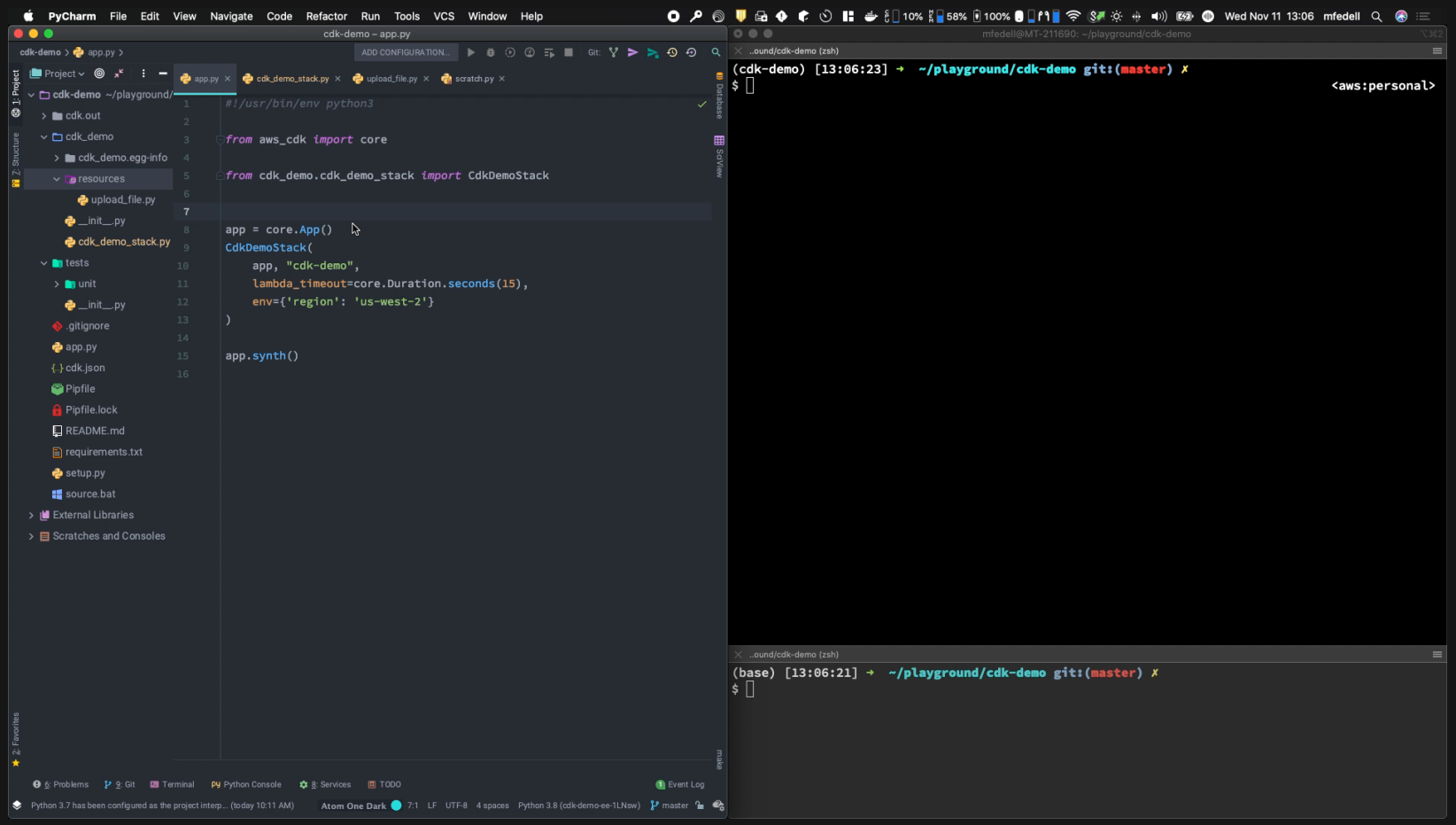
- Define cloud resources using a familiar syntax and paradigm
- Extensive IDE support to guide configuration
- Automatically default best-practice configuration
- Manage IAM permissions using least-privilege defaults
- Easy resource relationships: e.g. `my_bucket.grant_write(my_lambda)`
- Publish or consume useful patterns as common libraries
- Group and scope resources to manage complex applications
- Great CLI to monitor and manage deployments

A (hopefully) Convincing Example

- Roughly 13 lines of code produces:
  - 500+ lines of cloudformation defining:
  - 50 resources of:
  - 19 different types

```python
class MyEcsConstructStack(core.Stack):

    def __init__(self, scope: core.Construct, id: str, **kwargs) -> None:
        super().__init__(scope, id, **kwargs)

        vpc = ec2.Vpc(self, "MyVpc", max_azs=3)     # default is all AZs in region

        cluster = ecs.Cluster(self, "MyCluster", vpc=vpc)

        ecs_patterns.ApplicationLoadBalancedFargateService(self, "MyFargateService",
            cluster=cluster,            # Required
            cpu=512,                    # Default is 256
            desired_count=6,            # Default is 1
            task_image_options=ecs_patterns.ApplicationLoadBalancedTaskImageOptions(
                image=ecs.ContainerImage.from_registry("amazon/amazon-ecs-sample")),
            memory_limit_mib=2048,      # Default is 512
            public_load_balancer=True)  # Default is False
```

Quick “Live” Demo



Some Handy Resources

- CDK Workshop: Great tutorial straight from AWS
  - https://cdkworkshop.com/
- CDK Examples: Common usage for most services – from AWS
  - https://github.com/aws-samples/aws-cdk-examples
- CDK Documentation: TS API Spec (links to other languages)
  - https://docs.aws.amazon.com/cdk/api/latest/docs/aws-construct-library.html
- CDK Getting Started: Core concepts and AWS documentation
  - https://docs.aws.amazon.com/cdk/latest/guide/getting_started.html
- Additional Docs: AWS-endorsed links to other resources
  - https://docs.aws.amazon.com/cdk/latest/guide/home.html#additional_docs