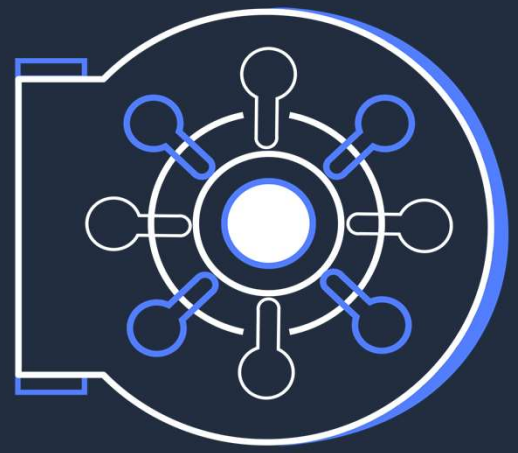
AWS TECHSHIFT

EMBARK



Infrastructure As Code

AWS TECHSHIFT Why IaC?

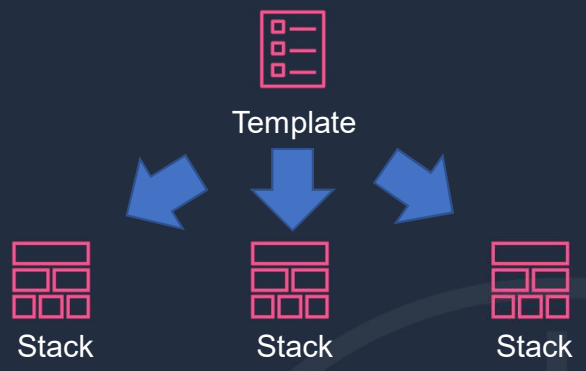
Cost

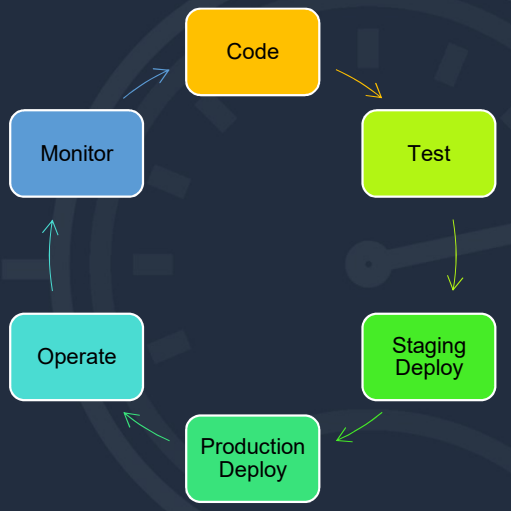
Speed

Risk

AWS TECHSHIFT Why IaC?

- Single source of truth to deploy the whole stack.
- Infrastructure that you can replicate, re-deploy, re-purpose
- Version control your infrastructure and your application together.
- Build your infrastructure and run it through your CI/CD pipeline.





AWS TECHSHIFT What is IaC?

Infrastructure as code (IaC) is the process of managing and provisioning computer data centers through machine-readable definition files, rather than physical hardware configuration or interactive configuration tools.

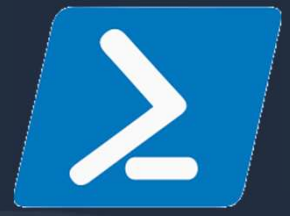
AWS TECHSHIFT Types of IaC

Declarative

Define how the system should be after the update. The system is responsible for defining the steps needed to reach that goal.

AWS CloudFormation

AWS Command Line Interface



Imperative

Define the specific steps to be executed on the system end with the desired conclusion.

AWS TECHSHIFT AWS CloudFormation

AWS TECHSHIFT JSON

JavaScript Object Notation

- Name-value pairs
- Similar to XML
- Executable by JavaScript Engine

```
{
"AWSTemplateFormatVersion" : "2010-09-09",
"Description" : "Create a Simple S3 bucket with parameter to choose own bucket name",
"Parameters": {
    "S3NameParam" : {
        "Type": "String",
        "Default" : "saurabh-dafaultbucket",
        "Description" : "Enter the Bucket Name",
        "MinLength" : "5",
        "MaxLength" : "30"
            }
        },

"Resources" : {
    "Bucket" : {
        "Type" : "AWS::S3::Bucket",
            "Properties" : {
                "AccessControl" : "PublicRead",
                "BucketName" : {"Ref" : "S3NameParam" },
                "Tags" : [ {"Key" : "Name" , "Value" : "MyBucket"} ]
                        }
                }
        },

"Outputs" : {
    "BucketName" : {
        "Description" : "BucketName" ,
        "Value" : { "Ref" : "S3NameParam"}
                }
            }

}
```

YAML Ain't Markup Language

- YAML is a human friendly data serialization standard
- Name-value pairs
- Comments: #
- ~~"{" and ";"~~
- Spacing is important

```yaml
Resources:
  DB:
    Type: "AWS::RDS::DBInstance"
    Properties:
      AllocatedStorage: 5
      StorageType: gp2
      DBInstanceClass: db.t2.micro
      DBName: wordpress
      Engine: MySQL
      MasterUsername: wordpress
      MasterUserPassword: w0rdpr355
  EC2:
    Type: AWS::EC2::Instance
    Properties:
      ImageId: ami-c481fad3 # N.Virginia - Ama Sept'16
      InstanceType: t2.micro
  S3:
    Type: "AWS::S3::Bucket"
    Properties:
      BucketName: wp-xxxxxx # replace xxxxxx with random
```

AWS TECHSHIFT Template Anatomy

1. Format Version
2. *Transform (new)*
3. Description
4. Metadata
5. Parameters
6. Mappings
7. Conditions
8. **Resources (required)**
9. Outputs

Format Version

- Currently only supports 1 value "2010-09-09"

Description

- Just a JSON string for you to provide note about the template

Parameters

- Validation of AWS resource types
- Input validation and restriction
- Occurs prior to stack creation
- Lists are possible for some type
- Control UI using AWS::CloudFormation::Interface

AWS::EC2::AvailabilityZone::Name

AWS::EC2::Image::Id

AWS::EC2::Instance::Id

AWS::EC2::KeyPair::KeyName

AWS::EC2::SecurityGroup::GroupName

AWS::EC2::SecurityGroup::Id

AWS::EC2::Subnet::Id

AWS::EC2::Volume::Id

AWS::EC2::VPC::Id

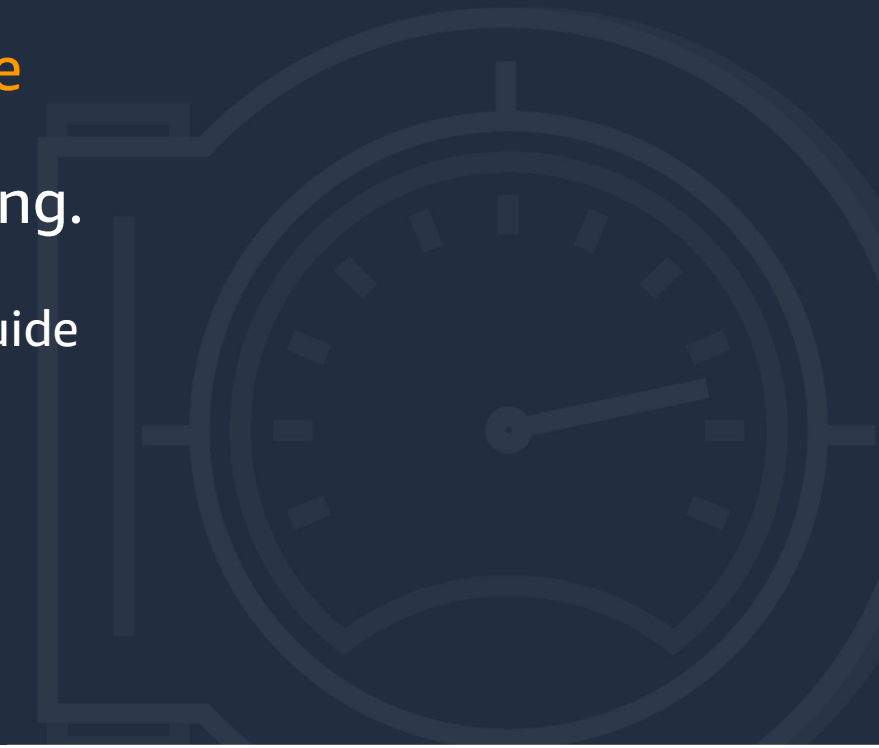
List<AWS::EC2::Subnet::Id>

Mappings

- Reference table, matches a key to a corresponding set of named values.
- Use Fn::FindInMap intrinsic function to retrieve values in a map.

Resources

- Only section that is not optional.
- Define AWS resources to create / update
- Supports 163 resource types, and growing.
  - Always refer back to CloudFormation user guide



Outputs

- Output values to view from the console or;
- Values returned from stack call
- Used with nested stack and cross stack references.
  - Pass parameter values from one stack to another.
- Output for stack, Export for global

AWS TECHSHIFT Template Anatomy

Intrinsic functions

Fn::Base64
Fn::FindInMap
Fn::GetAtt
Fn::GetAZs
Fn::Join
Fn::Select
Ref

Pseudo parameters

AWS::NotificationARNs
AWS::Region
AWS::StackId
AWS::StackName

AWS TECHSHIFT Template Anatomy

```
AWSTemplateFormatVersion: 2010-09-09
Description: CloudFormation for TechShift Modernise - Photo Gallery
Parameters:
  KeyName:
    Description: Name of an existing EC2 KeyPair to enable SSH access to the instance
    Type: AWS::EC2::KeyPair::KeyName
    ConstraintDescription: can contain only ASCII characters.

  SSHLocation:
    Description: The IP address range that can be used to SSH to the EC2 instances
    Type: String
    MinLength: '9'
    MaxLength: '18'
    AllowedPattern: '(\d{1,3})\.(\d{1,3})\.(\d{1,3})\.(\d{1,3})/(\d{1,2})'
    ConstraintDescription: must be a valid IP CIDR range of the form x.x.x.x/x.
    Default: 0.0.0.0/0

Mappings:
  RegionMap:
    ca-central-1:
      "AMI": ami-03338e1f67dae0168
    us-east-2:
      "AMI": ami-02bcbb802e03574ba
    eu-west-3:
```

AWS TECHSHIFT Template Anatomy

```yaml
Resources:
  WebServer:
    Type: AWS::EC2::Instance
    Properties:
      ImageId: !FindInMap [RegionMap, !Ref "AWS::Region", AMI]
      InstanceType: t3.medium
      IamInstanceProfile: !Ref DeployRoleProfile
      KeyName: !Ref KeyName
      NetworkInterfaces:
        - AssociatePublicIpAddress: true
          DeviceIndex: 0
          GroupSet:
            - Ref: PublicSecurityGroup
          SubnetId:
            Ref: PublicSubnetA
      Tags:
        - Key: 'Name'
          Value: !Join ['', [!Ref 'AWS::StackName', '::WebServer'] ]
      UserData:
        Fn::Base64:                              # YAML makes userdata much cleaner
          !Sub |
              #!/bin/bash
              echo ==== Starting UserData Script ====
              curl -k -o /root/setup.sh http://d3eglt6sb590rd.cloudfront.net/assets/setup.sh
              chmod +x /root/setup.sh
              sudo -i /root/setup.sh
              echo ==== Finished UserData Script ====
```

AWS TECHSHIFT Template Anatomy

```
Outputs:
  URL:
    Value:
      Fn::Join:
      - ''
      - - http://
        - Fn::GetAtt:
          - WebServer
          - PublicIp
    Description: Lab 1 application URL
    Export:
      Name: "TSAGallery-ServerURL"
```

End