



An evening in Toulouse with AWS CloudFormation



CloudFormation

Julien Simon
Principal Technical Evangelist, AWS
[@aws_actus](#) [@julsimon](#)



**Toulouse
DevOps**

24/02/2016


You're more than welcome to tweet about this presentation

Pictures too 😊

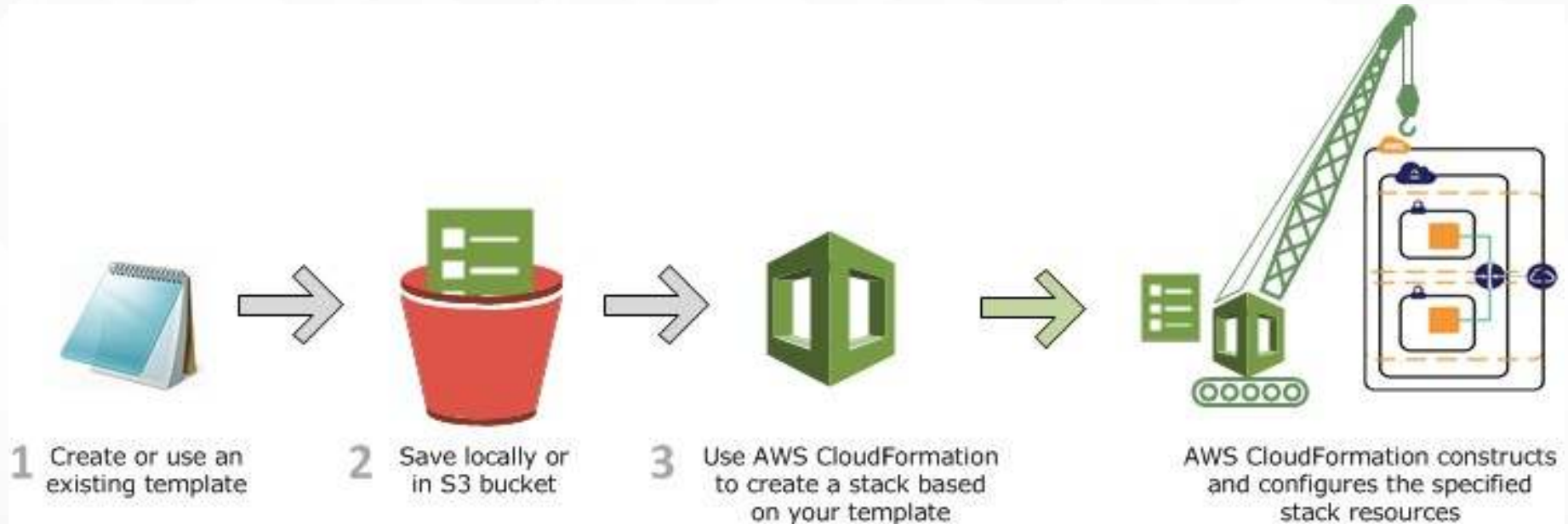
@julsimon @aws_actus @ToulouseDevops

#aws #cloudformation

AWS CloudFormation

- Fundamental service in AWS used for automating deployment and configuration of resources
- CloudFormation Template 
 - JSON-formatted document which describes a configuration to be deployed in an AWS account
 - When deployed, refers to a “stack” of resources
 - Not a “script”, a *document*

AWS CloudFormation





```

"Conditions": {
  "HaveNoOtherRoles": { "Fn::Equals": [{ "Ref": "OtherRoles" }, ""] },
  "HaveEbs": { "Fn::Not": [{ "Fn::Equals": [{ "Ref": "EbsVolumeSize" }, "0"] } ] },
  "HaveEbsSnapshotId": { "Fn::Not": [{ "Fn::Equals": [{ "Ref": "EbsSnapshotId" }, ""] } ] },
  "HaveAdditionalTagKey": { "Fn::Not": [{ "Fn::Equals": [{ "Ref": "AdditionalTagKey" }, ""] } ] },
  "HaveAdditionalTagValue": { "Fn::Not": [{ "Fn::Equals": [{ "Ref": "AdditionalTagValue" }, ""] } ] },
  "HaveSSL": { "Fn::Not": [{ "Fn::Equals": [{ "Ref": "SSLPort" }, "0"] } ] },
  "IsHTTP": { "Fn::Equals": [{ "Ref": "ElbProtocol" }, "HTTP" ] },
  "HaveSpotPrice": { "Fn::Not": [{ "Fn::Equals": [{ "Ref": "SpotPrice" }, ""] } ] }
},
"Resources": {
  "AutoScalingGroup": {
    "Type": "AWS::AutoScaling::AutoScalingGroup",
    "UpdatePolicy": {
      "AutoScalingRollingUpdate": {
        "MaxBatchSize": "1",
        "MinInstancesInService": "0",
        "PauseTime": "PT15M",
        "WaitOnResourceSignals": "true"
      }
    },
    "Properties": {
      "LaunchConfigurationName": { "Ref": "LaunchConfig" },
      "LoadBalancerNames": [ { "Ref": "ElasticLoadBalancer" } ],
      "MinSize": { "Ref": "MinPoolSize" },
      "MaxSize": { "Ref": "MaxPoolSize" },
      "AvailabilityZones": { "Fn::FindInMap": [ "AZConfig", "AvailabilityZones", "all" ] },
      "VPCZoneIdentifier": { "Ref": "EC2SubnetsIds" },
      "Tags": [
        { "Fn::If": [
          "HaveAdditionalTagKey",
          {
            "Key": { "Ref": "AdditionalTagKey" },
            "Value": {
              "Fn::If": [
                "HaveAdditionalTagValue",
                { "Ref": "AdditionalTagValue" },
                ""
              ]
            }
          },
          "PropagateAtLaunch": "true"
        ],
        { "Ref": "AWS::NoValue" }
      ]
    },
    "Key": "Name", "Value": { "Fn::Join": [ ".", [ { "Ref": "ServiceName" }, { "Ref": "EnvironmentName" } ] ] },
    "Key": "cost", "Value": { "Ref": "Cost" }, "PropagateAtLaunch": "true",
    "Key": "environment", "Value": { "Ref": "EnvironmentName" }, "PropagateAtLaunch": "true"
  }
}

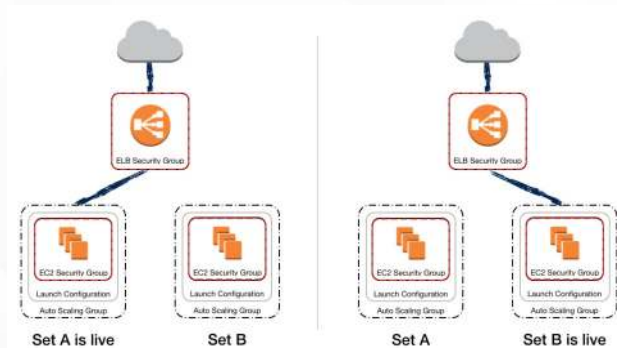
```

Infrastructure as code

- Versioned, auditable blueprints (developers can contribute)
- Quick to deploy, repeatable, tested infrastructure
- Enables CI/CD for infrastructure (just like everything else)
- Deploy many times, anywhere

Typical use cases for AWS CloudFormation

- Used internally by many AWS products (Elastic Beanstalk, ECS)
- Replicating environments
 - dev, integration, pre-production, production
 - Same architecture, different sizing → template + parameters
- Deploying in a different region
- Green / blue deployments
- Disaster Recovery



Case study: Viadeo



<https://www.youtube.com/watch?v=JJm4V5fd0Z8>

CloudFormation Template Structure

```
{
  "AWSTemplateFormatVersion" : "version date",

  "Description" : "JSON string",

  "Metadata" : {
    template metadata
  },

  "Parameters" : {
    set of parameters
  },

  "Mappings" : {
    set of mappings
  },

  "Conditions" : {
    set of conditions
  },

  "Resources" : {
    set of resources
  },

  "Outputs" : {
    set of outputs
  }
}
```

Resources

Describe detailed configuration of a resource in AWS

Include, but not limited to:

- IAM Policies, Users, Groups, Roles
- VPCs, Subnets, NACLs, Security Groups
- EC2 instances, AutoScaling Groups
- RDS Databases, S3 Buckets
- Elastic Load Balancers
- CloudWatch Alarms
- Lambda Functions
- Logging (CloudTrail, CW Logs)

```
"sysadminPolicy" : {
  "Type" : "AWS::IAM::ManagedPolicy",
  "Properties" : {
    "PolicyDocument" :
      {
        "Version": "2012-10-17",
        "Statement": [
          {
            "Effect": "Allow",
            "NotAction": "iam:*",
            "Resource": "*"
          },
          {
            "Effect": "Deny",
            "Action": "aws-portal:*Billing",
            "Resource": "*"
          },
          {
            "Effect" : "Deny",
            "Action" : [ "cloudtrail:DeleteTrail",
                        "cloudtrail:StopLogging",
                        "cloudtrail:UpdateTrail" ],
            "Resource" : "*"
          }
        ]
      },
    "Roles" : [
      { "Ref" : "sysadminRole" }
    ],
    "Groups" : [
      { "Ref" : "sysadminGroup" }
    ]
  }
},
```

Nested Templates

- CloudFormation stacks themselves can be resources

```
"AWS::CloudFormation::Stack"
```

- Useful for making reusable templates, segmenting resources, and avoiding template size limitations
- Launching a template with nested stacks will launch multiple sub-stacks
- Deleting the launching stack will, by default, delete all substacks

Logical ID	Physical ID	Type	Status
stack1	arn:aws:cloudformation:us-east-1:979676883363:stack/GoldBase1-stack1-11QK6Q0K6AZD5/8cc9fb90-78d6-11e5-ab62-5001ba48c2d2	AWS::CloudFormation::Stack	CREATE_COMPLETE
stack2	arn:aws:cloudformation:us-east-1:979676883363:stack/GoldBase1-stack2-32N9A77OO46U/8d192d00-78d6-11e5-a764-50e2416294a8	AWS::CloudFormation::Stack	CREATE_COMPLETE
stack3	arn:aws:cloudformation:us-east-1:979676883363:stack/GoldBase1-stack3-10QEXK61Z61LP/46c9780-78d6-11e5-86e1-50e24162947c	AWS::CloudFormation::Stack	CREATE_COMPLETE
stack4	arn:aws:cloudformation:us-east-1:979676883363:stack/GoldBase1-stack4-1CIRM21C3IQ5G/2570cf40-78d7-11e5-abcb-507bb903ae0a	AWS::CloudFormation::Stack	CREATE_COMPLETE

Parameters

- Used to pass in variables when launching a stack
- Use the “Ref” function to reference these variables in the Resources section of the template

```
"Parameters" : {  
  "InstanceTypeParameter" : {  
    "Type" : "String",  
    "Default" : "t1.micro",  
    "AllowedValues" : ["t1.micro", "m1.small", "m1.large"],  
    "Description" : "Enter t1.micro, m1.small, or m1.large. Default is t1.micro."  
  }  
}
```

```
"Ec2Instance" : {  
  "Type" : "AWS::EC2::Instance",  
  "Properties" : {  
    "InstanceType" : { "Ref" : "InstanceTypeParameter" },  
    "ImageId" : "ami-2f726546"  
  }  
}
```

Mappings

- Provides a set of custom named-value pairs
- Use for setting values based on different possible conditions (most notably, regions)
- Commonly used for mapping different AMI IDs to make template reusable across multiple AWS regions
- Use the FindInMap function when referencing in resources

```
"Mappings" : {  
  "RegionMap" : {  
    "us-east-1" : {  
      "AMI" : "ami-76f0061f"  
    },  
    "us-west-1" : {  
      "AMI" : "ami-655a0a20"  
    },  
    "eu-west-1" : {  
      "AMI" : "ami-7fd4e10b"  
    },  
    "ap-southeast-1" : {  
      "AMI" : "ami-72621c20"  
    },  
    "ap-northeast-1" : {  
      "AMI" : "ami-8e08a38f"  
    }  
  }  
},
```

```
"ImageId" : { "Fn::FindInMap" : [ "RegionMap", { "Ref" : "AWS::Region" }, "AMI" ] }
```

Conditions

- Allow you to determine if a resource gets created or a property is defined
- The “Condition” attribute applied to any resource to specify a condition defined in the “Conditions” section of the template
- Condition must evaluate to true, otherwise the resource will not get created

```
"Parameters" : {
  "EnvType" : {
    "Description" : "Environment type.",
    "Default" : "test",
    "Type" : "String",
    "AllowedValues" : ["prod", "test"],
    "ConstraintDescription" : "must specify prod or test."
  },
},
"Conditions" : {
  "CreateProdResources" : {"Fn::Equals" : [{"Ref" : "EnvType"}, "prod"]}
},
```

```
"MountPoint" :
{
  "Type" : "AWS::EC2::VolumeAttachment",
  "Condition" : "CreateProdResources",
  "Properties" : {
    "InstanceId" : { "Ref" : "EC2Instance" },
    "VolumeId" : { "Ref" : "NewVolume" },
    "Device" : "/dev/sdh"
  }
}
```

AWS CloudFormation best practices

- Don't start from scratch 😊
- Read sample templates
- Use AWS CloudFormer as a starting point
- Reuse as much as possible
- Don't go crazy on nested stacks... 1 level should be enough
- Use parameters: environment, region, instance names, instance sizes, etc..
- Tag everything!

Let's summon the clouds!

*Demo gods, I'm your humble servant,
please be good to me*

Create, update, delete a basic stack

Create a VPC with 4 subnets

Create A LAMP stack

Create an app in Elastic Beanstalk and look
at the template

Create an ECS cluster and look at the
template



@aws_actus @julsimon
www.facebook.com/groups/AWSFrance/

AWSome Day 
aws.amazon.com/fr/awssomeday/



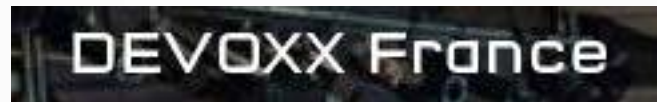
March 7-8



March 23-24



April 6-7 (Lyon)



April 20-22



April 25



AWS Summit
May 31st



AWS User Groups in Paris,
Lyon, Nantes, Lille & Rennes
(meetup.com)



BONUS SLIDES

AWS CloudFormation resources

Documentation

<https://aws.amazon.com/fr/documentation/cloudformation/>

https://docs.aws.amazon.com/fr_fr/AWSCloudFormation/latest/UserGuide/cfn-sample-templates.html

Blogs

<https://aws.amazon.com/fr/blogs/aws/category/aws-cloud-formation/>

<https://blogs.aws.amazon.com/application-management/blog/tag/CloudFormation>

Sessions @ AWS re:Invent 2015

ARC307 - Infrastructure as Code: [slides](#) and [video](#)

ARC401 - Cloud First: New Architecture for New Infrastructure: [slides](#) and [video](#)

DVO303 - Scaling Infrastructure Operations with AWS: [slides](#) and [video](#)

DVO304 - AWS CloudFormation Best Practices: [slides](#) and [video](#)

DVO310 - Benefit from DevOps When Moving to AWS for Windows: [slides](#) and [video](#)

DVO401 - Deep Dive into Blue/Green Deployments on AWS: [slides](#) and [video](#)

SEC312 - Reliable Design and Deployment of Security and Compliance: [slides](#) and [video](#)

Create a Git repository with AWS CodeCommit

```
$ aws codecommit create-repository  
--repository-name cfdemo --region us-east-1  
--repository-description "CloudFormation  
demo"
```

```
$ git clone ssh://git-codecommit.us-  
east-1.amazonaws.com/v1/repos/cfdemo
```

Managing AWS CloudFormation with the CLI

```
$ aws cloudformation validate-template --template-body  
file://template.json
```

```
$ aws cloudformation create-stack --template-body  
file://template.json --stack-name MyTemplate --region eu-  
west-1
```

```
$ aws cloudformation get-template --stack-name MyTemplate
```

```
$ aws cloudformation update-stack --stack-name MyTemplate  
--template-body file://template.json
```

```
$ aws cloudformation delete-stack --stack-name MyTemplate
```