
AWS
re:Invent
NOV. 28 – DEC. 2, 2022 | LAS VEGAS, NV

BOA403

Extend AWS CloudFormation and AWS CDK with third-party resources

Matteo Rinaudo (he/him)

Sr. Developer Advocate, AWS CloudFormation
Amazon Web Services

Mircea Livadariu (he/him)

Principal Engineer, AWS CloudFormation
Amazon Web Services

Extend CloudFormation

The AWS CloudFormation registry

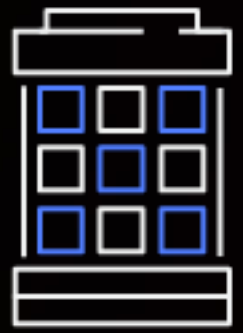
AWS CloudFormation

Modules

Resource types

Hooks

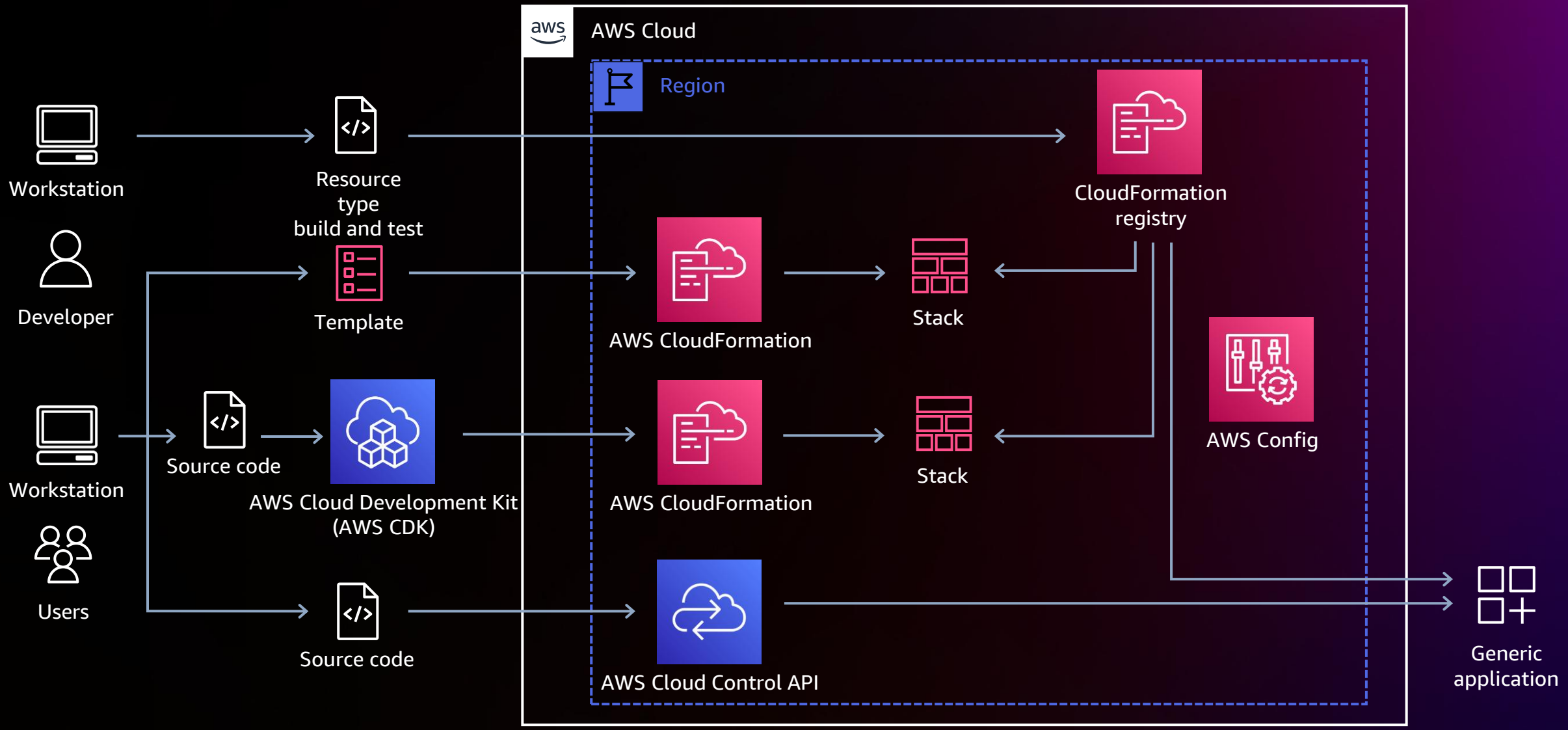
Submitting and publishing



Private extensions

Public extensions

Developing and consuming resource types



Software development life cycle (SDLC)

Use case

CFN-CLI and Java, Python, TypeScript, or Go

Model and develop your resource

Run unit and contract tests

Submit your resource to the registry

Bootstrap your project

```
$  cfn init
Initializing new project
Do you want to develop a new resource(r) or a module(m) or a hook(h)?.
>> r
What's the name of your resource type?
(Organization::Service::Resource)
>> AWSSamples::EC2::ImportKeyPair
Select a language for code generation:
[1] go
[2] java
[3] python36
[4] python37
[5] typescript
(enter an integer):
>> 4
Use docker for platform-independent packaging (Y/n)?
```

Model your resource with the schema

```
{
    "typeName": "AWSSamples::EC2::ImportKeyPair",
    "description": "Sample resource schema demonstrating a model for an example
resource type to import a key pair.",
    "sourceUrl": "https://github.com/aws-cloudformation/aws-cloudformation-
samples.git",
    "definitions": {
        "Tag": {
            "description": "A key-value pair to associate with a resource.",
            "type": "object", [...]
     "properties": {
        "Tags": {
            "description": "An array of key-value pairs to apply to the resource.",
            "type": "array", [...]
            "items": {
                "$ref": "#/definitions/Tag" [...]
```

Main property attributes

primaryIdentifier	createOnlyProperties	readOnlyProperties	writeOnlyProperties
Uniquely identifies the resource	User-specified; resource creation only, not updates	Returned with List and Read; not user-provided	Not returned with List and Read; user-provided; typically, sensitive values

AWSSamples::EC2::ImportKeyPair: uses Amazon EC2's ImportKeyPair API:

KeyFingerprint: generated at runtime; e.g., MD5 public key fingerprint for RSA keys

KeyName: user input; to change its value, you need to create another resource

KeyPairId: generated at runtime; e.g., "key-01234abcd [...]"

KeyType: available at runtime via Boto 3's describe_key_pairs() response; e.g., "rsa"

PublicKeyMaterial: like KeyName; also, not returned with API calls

primaryIdentifier?	createOnlyProperties?	readOnlyProperties?	writeOnlyProperties?
KeyPairId	KeyName PublicKeyMaterial	KeyPairId KeyFingerprint KeyType	PublicKeyMaterial

CRUDL handlers

Create Read Update Delete List

Handlers sections in the schema

```
"handlers": {
    "create": {
        "permissions": [
            "ec2:ImportKeyPair",
            "ec2:CreateTags"
        ]
    },
    "read": {
        "permissions": [
            "ec2:DescribeKeyPairs"
        ]
    },
    "update": {
        "permissions": [
            "ec2:CreateTags",
            "ec2:DeleteTags"
        ]
    },
    "delete": {
        "permissions": [
            "ec2:DeleteKeyPair",
            "ec2:DescribeKeyPairs"
        ]
    },
    "list": {
        "permissions": [
            "ec2:DescribeKeyPairs"
        ]
    }
} [...]
```

Example create handler snippet

```
@resource.handler(Action.CREATE)
def create_handler(
    session: Optional[SessionProxy],
    request: ResourceHandlerRequest,
    callback_context: MutableMapping[str, Any],
) -> ProgressEvent:
[...]
    try:
        client = _get_session_client(
            session,
            "ec2",
        )
        [...]
        response = client.import_key_pair(
            **import_key_pair_kwargs,
        )
        [...]
```

ProgressEvent examples

```
@resource.handler(Action.CREATE)
def create_handler(
[...]

    return ProgressEvent.failed(
        # CloudFormation handler error code
        handler_error_code,
        # Message, e.g., exception message
        f"Error: {error_message}",
    )
```

```
# Create, Read, Update handlers
return ProgressEvent(
    status=OperationStatus.SUCCESS,
    resourceModel=model,
)

# List handler
return ProgressEvent(
    status=OperationStatus.SUCCESS,
    resourceModels=models,
)

# Delete handler
return ProgressEvent(
    status=OperationStatus.SUCCESS,
)
```

Mapping service API errors to handler errors

```
    # Error codes for the Amazon EC2 API:
    # https://docs.aws.amazon.com/AWSEC2/latest/APIReference/errors-overview.html
    if api_error_code == "InvalidKeyPair.NotFound":
        return HandlerErrorCode.NotFound
    elif api_error_code == "InvalidKeyPair.Duplicate":
        return HandlerErrorCode.AlreadyExists
    elif api_error_code in [
        "InvalidKey.Format",
        "InvalidKeyPair.Format",
        [...]
    ]:
        return HandlerErrorCode.InvalidRequest
[...]
```

Example exception and return blocks

```
[...]
    except botocore.exceptions.ClientError as ce:
        return _progress_event_failed(
            handler_error_code=_get_handler_error_code(
                ce.response["Error"]["Code"],
            ),
            error_message=str(ce),
            traceback_content=traceback.format_exc(),
        )
[...]
```

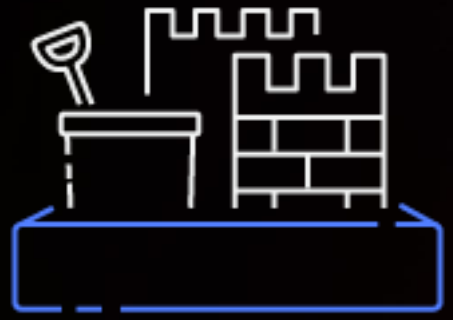
Resource types and coding best practices

Reuse and correlate with $ref in schema

Map API errors to handler errors

Document code and usage

Testing resource types



Unit tests

Contract tests

Running contract tests

Terminal Window 1: Lambda endpoint mocked locally

```
$  sam local start-lambda
Starting the Local Lambda Service. You can now invoke your Lambda Functions
defined in your template through the endpoint.
2022-10-21 12:00:01  * Running on http://127.0.0.1:3001/ (Press CTRL+C to quit)
```

Terminal Window 2: run contract tests; real API calls made on your account

```
$  cfn generate && cfn submit --dry-run && cfn test
```

Contract test inputs

Example for "create": /your-project-root/inputs/inputs_1_create.json:

```
{
    "KeyName": "example-keypair-for-contract-tests-1",
    "PublicKeyMaterial": "{{KeyPairPublicKeyForContractTests}}" [...]
```

Example for "update": /your-project-root/inputs/inputs_1_update.json:

```
{
    "KeyName": "example-keypair-for-contract-tests-1",
    "PublicKeyMaterial": "{{KeyPairPublicKeyForContractTests}}" [...]
```

Example for "invalid": /your-project-root/inputs/inputs_1_invalid.json:

```
{
    "KeyName": "example-keypair-for-contract-tests-1",
    "PublicKeyMaterial": "invalid" [...]
```

Contract tests best practices

Start contract testing early in the SDLC

Start with one contract test first; e.g. , `contract_create_delete`

Choose the long traceback option type for debugging

```
$  cfn test -- -k contract_create_delete
```

```
$  cfn test -- --tb=long
```

Submitting resource types to the registry

CFN-CLI

CloudFormation
stack or StackSet

CloudFormation CLI submission option

```
$ cfn submit --region us-east-1 --set-default
```

CFN-CLI

Developer

AWS CloudFormation registry

Stack

AWS Identity and Access Management (IAM)

Role

Stack or StackSets submission method

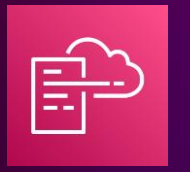
```
$  cfn submit --dry-run
```

CFN-CLI

Resource project package

Template

Stack
(or StackSet)



AWS CloudFormation registry

```
AWS::CloudFormation::ResourceVersion
   + SchemaHandlerPackage S3 URL,
AWS::CloudFormation::ResourceDefaultVersion,
AWS::Logs::LogGroup,
AWS::IAM::Role (ExecutionRole),
AWS::IAM::Role (LogAndMetricsDeliveryRole),
```

Developer

S3 bucket with resource project package

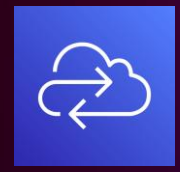
Consuming resource types

AWS CloudFormation

Template

AWS Cloud Development Kit
(AWS CDK)

Source code



AWS Cloud Control API

Source code

CloudFormation template option

```
[...]
Resources:
  KeyPair:
    Type: AWSSamples::EC2::ImportKeyPair
    Properties:
      KeyName: !Ref 'KeyPairName'
      PublicKeyMaterial: !Ref 'KeyPairPublicKey'
      Tags:
        - Key: Name
          Value: !Ref 'KeyPairName'
        - Key: OrganizationName
          Value: !Ref 'OrganizationName'
        - Key: OrganizationBusinessUnitName
          Value: !Ref 'OrganizationBusinessUnitName'
```

```
Outputs:
  KeyFingerprint:
    Description: MD5 public key fingerprint.
    Value: !GetAtt KeyPair.KeyFingerprint

  KeyPairId:
    Description: The ID of the key pair.
    Value: !Ref 'KeyPair'

  KeyType:
    Description: The type of the key pair.
    Value: !GetAtt KeyPair.KeyType
```

AWS CDK option: Construct importing

Open source tool: cdk-import

Features include CDK construct generation from resource types

Consume the imported resource from your project

```
$  npm install -g cdk-import

$  cd my-project/

$  curl -s https://raw.githubusercontent.com/aws-cloudformation/aws-cloudformation-samples/main/resource-types/awssamples-ec2-importkeypair/python/awssamples-ec2-importkeypair.json -o awssamples-ec2-importkeypair.json

$  cdk-import cfn -l python AWSSamples::EC2::ImportKeyPair --schema-file awssamples-ec2-importkeypair.json -o .
```

AWS CDK: Consume the imported construct

```
from aws_cdk import (
    App,
    Stack,
)
from constructs import Construct

from awssamples_ec2_importkeypair import CfnImportKeyPair


class CfnImportKeyPairStack(Stack):

    def __init__(self, scope: Construct, id: str, **kwargs) -> None:
        super().__init__(scope, id, **kwargs)
        CfnImportKeyPair(
            self, "MyImportedKeyPair", key_name="[...YOUR KEY NAME...]",
            public_key_material="[...YOUR PUBLIC KEY MATERIAL...]",
        )

app = App()
CfnImportKeyPairStack(app, "CfnImportKeyPairStack",)

app.synth()
```

AWS Cloud Control API option

Consistent low-level API consumption experience

Specify a resource type and a desired state

```
$  aws cloudcontrol create-resource \
    --type-name AWSSamples::EC2::ImportKeyPair \
    --desired-state '{"KeyName": "Example", "PublicKeyMaterial": " [...]"}'


$  aws cloudcontrol list-resources --type-name AWSSamples::EC2::ImportKeyPair
{
    "ResourceDescriptions": [
        {
            "Identifier": " [...]
    ],
    "TypeName": "AWSSamples::EC2::ImportKeyPair"
}
```

Call to action

AWS CloudFormation
community registry extensions repo

Discord server for
AWS CloudFormation

Q&A and resources

AWS CloudFormation workshop: Resource types lab

AWSSamples::EC2::ImportKeyPair example resource type in Python

Thank you!

Matteo Rinaudo

Mastodon:
@mrinaudo@awscommunity.social

@mrinaudo

Mircea Livadariu

Mastodon:
@mircea@awscommunity.social

Please complete the session survey in the **mobile app**