

AWS re:Invent

NOV. 28 – DEC. 2, 2022 | LAS VEGAS, NV

DOP401-R

Get better at building AWS CDK constructs

Kendra Neil

Software Development Engineer
Amazon Web Services

Mike Apted

Senior Manager
Amazon Web Services



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Agenda

Introduction to constructs

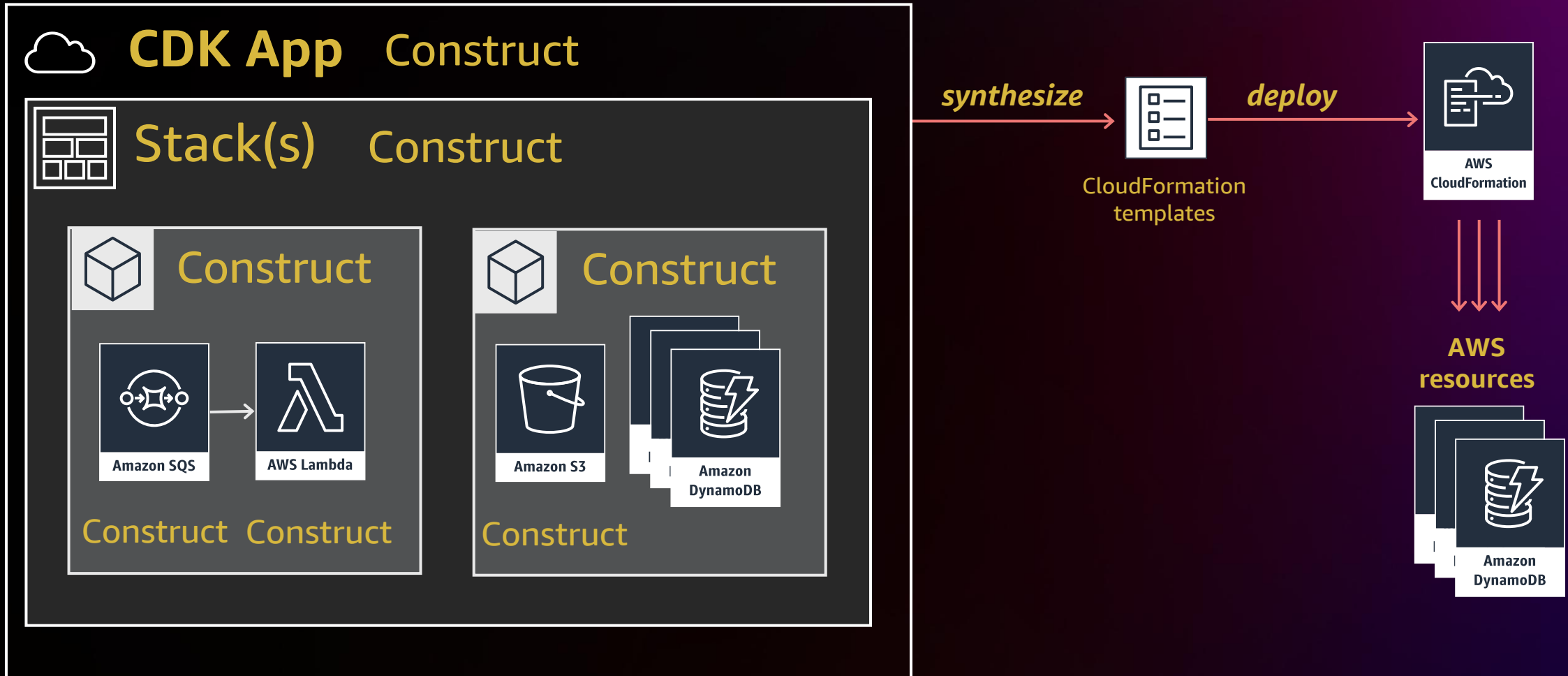
Best practices

Let's build!



Introduction to constructs

What is a construct?



What is in a construct?

Customization

```
export interface SimpleConstructProps {  
  readonly fifoQueue: boolean;  
}
```

```
export class SimpleConstruct extends Construct {  
  public readonly queue: Queue;
```

Customization

State guarantees

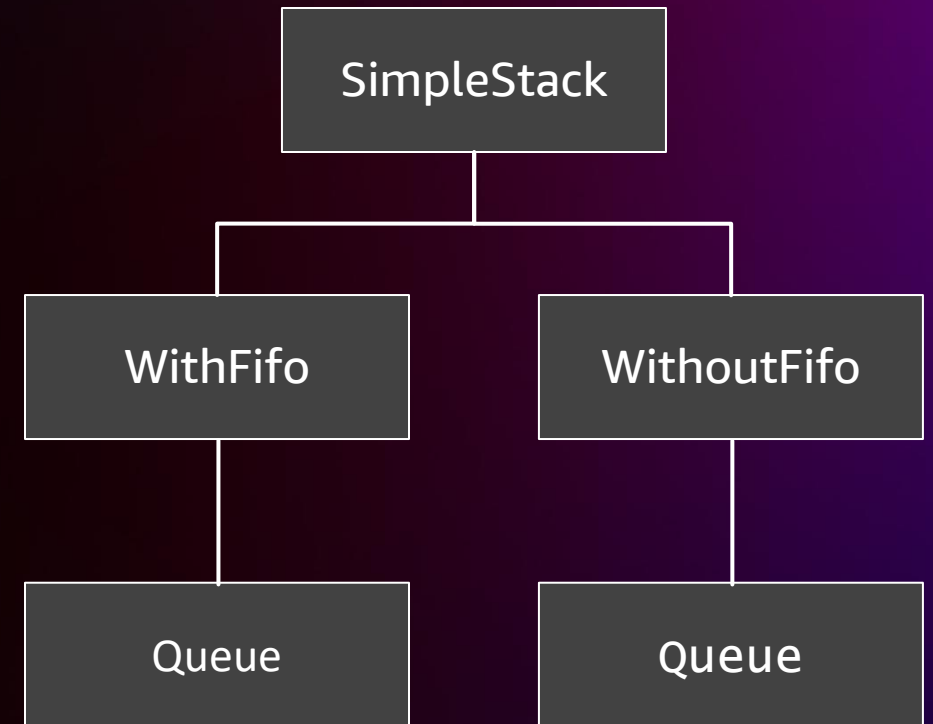
```
  constructor(scope: Construct, id: string, props: SimpleConstructProps) {  
    super(scope, id);
```

Tree building on construction

```
    this.queue = new Queue(this, 'Queue', {  
      fifo: props.fifoQueue,  
    });  
  }  
}
```

What is in a construct?

```
export class SimpleStack extends Stack {  
  constructor(scope: Construct, id: string) {  
    super(scope, id);  
  
    new SimpleConstruct(this, 'WithFifo', {  
      fifoQueue: true,  
    });  
  
    new SimpleConstruct(this, 'WithoutFifo', {  
      fifoQueue: false,  
    });  
  }  
}
```



The construct tree

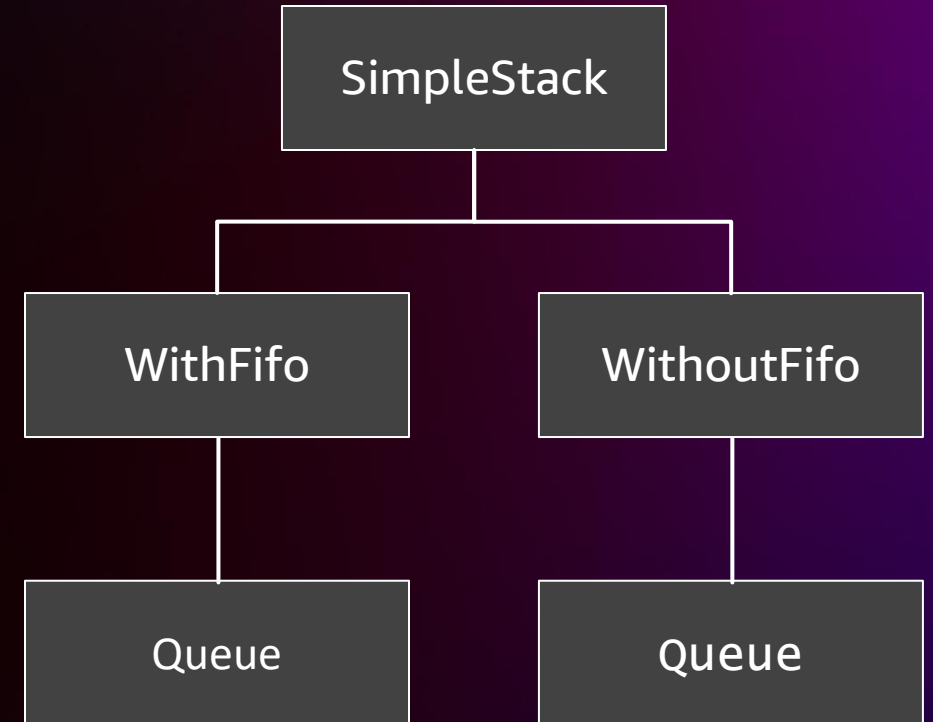
Resources:

withFifoQueue3027818E

Type: AWS::SQS::Queue

withoutFifoQueue3E08D69B

Type: AWS::SQS::Queue



The construct tree – The node attribute

node.children	The direct children of the construct
node.id	The identifier of the construct within its scope
node.path	The full path of the construct including the IDs of all of its parents
node.root	The root of the construct tree (the app)
node.scope	The scope (parent) of the construct, or undefined if the node is the root
node.scopes	All parents of the construct, up to the root
node.uniqueId	The unique alphanumeric identifier for this construct within the tree (by default, generated from node.path and a hash)

Construct levels

L3

Purpose-built constructs

Opinionated abstractions

L2

AWS constructs

High-level service constructs

L1

CloudFormation resources

Automatically generated

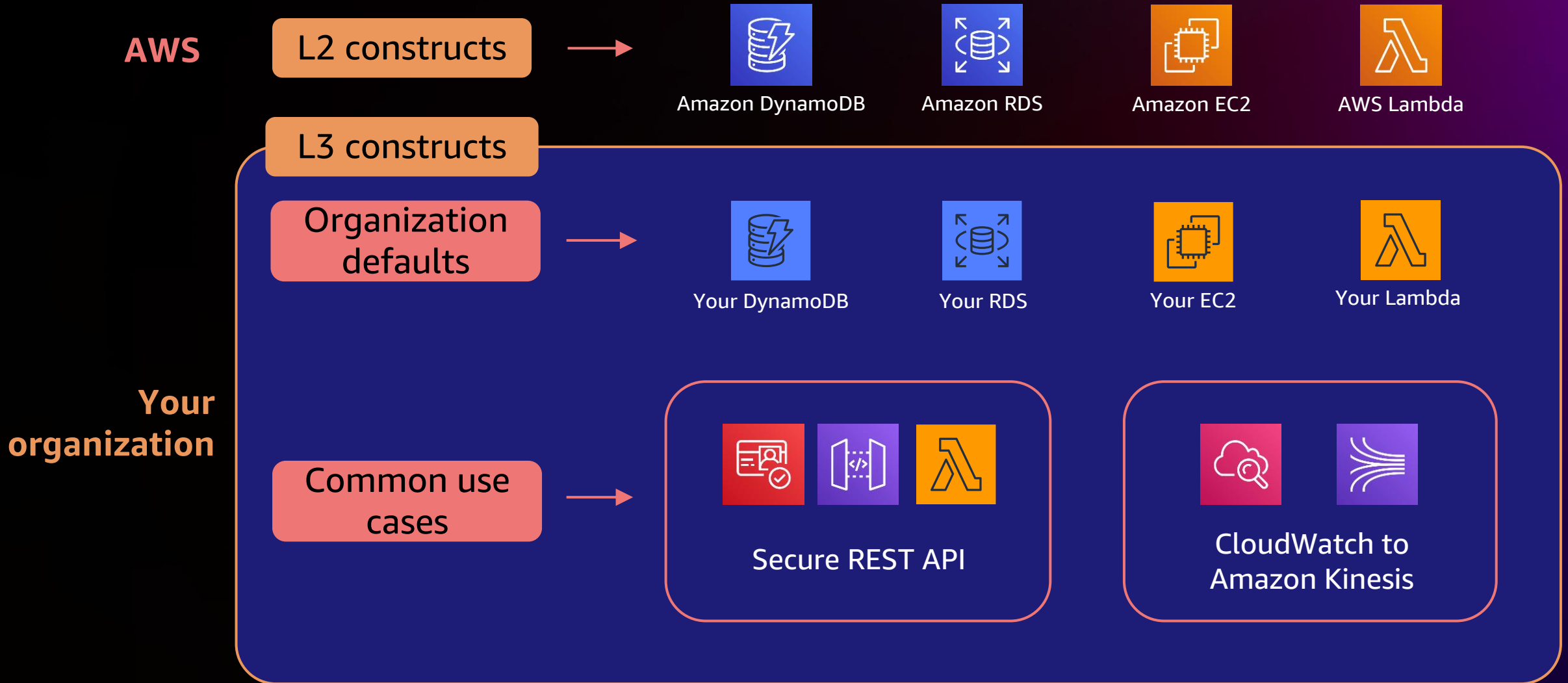
Custom resources

```
const onEvent: lambda.Function;  
const isComplete: lambda.Function;  
const myRole: iam.Role;  
  
const provider = new cr.Provider(this, 'MyProvider', {  
    onEventHandler: onEvent,  
    isCompleteHandler: isComplete, // optional async "waiter"  
    logRetention: logs.RetentionDays.ONE_DAY, // default is INFINITE  
    role: myRole, // to be assumed by the `lambda.amazonaws.com` service principal  
});  
  
new CustomResource(this, 'Resource1', { serviceToken: provider.serviceToken });  
new CustomResource(this, 'Resource2', { serviceToken: provider.serviceToken });
```

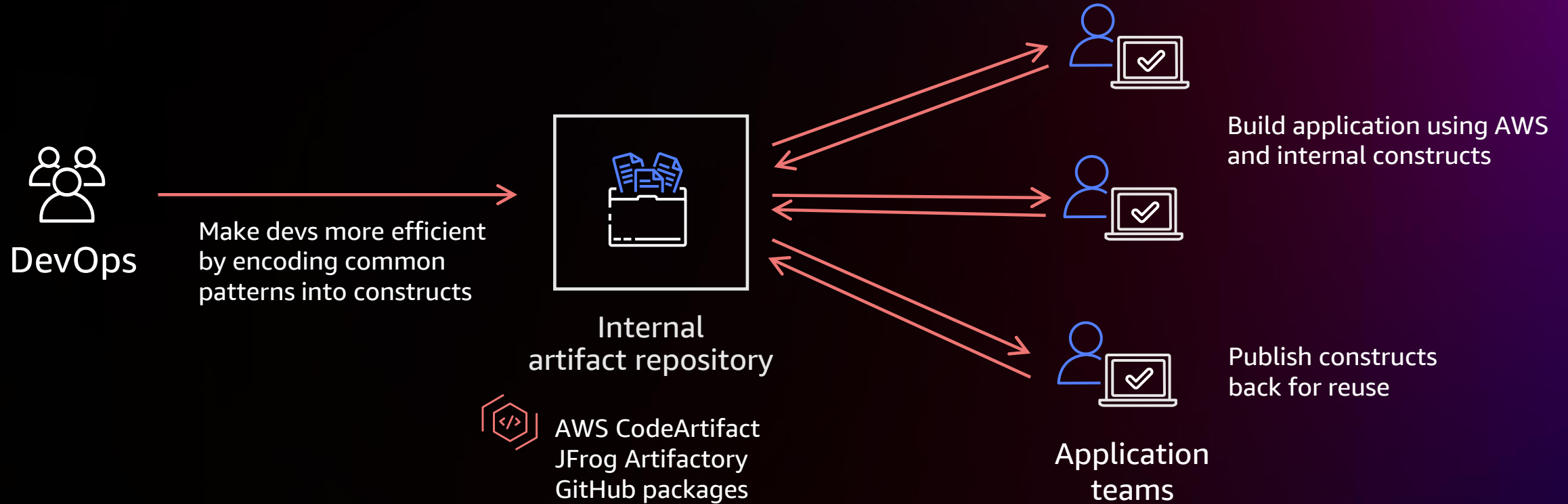
Best practices



Construct libraries



Construct libraries



Construct libraries

Do

- ✓ Write and publish using **jsii**
- ✓ Vend constructs
- ✓ Configure with properties
- ✓ Unit test your infrastructure
- ✓ Use constructs for convenience

Don't

- ✗ Vend stacks
- ✗ Read environment variables
- ✗ Change **ids** of stateful resources
- ✗ Use constructs for compliance

CDK applications

Do

- ✓ Parameterize your app for generated resource names
- ✓ Separate into stacks because of deployment properties
- ✓ Make your app deterministic (`cdk.context.json`)
- ✓ Allow CDK to manage roles
- ✓ Model all production stages

Don't

- ✗ Rely on physical names
- ✗ Separate because it *feels* like you should
- ✗ Do network lookups
- ✗ Import precreated roles
- ✗ Parameterize app to be a stage

Testing

- Snapshot tests

Test the synthesized AWS CloudFormation template against a previously stored baseline template

- Fine-grained assertions

Test specific aspects of the generated AWS CloudFormation template, such as "this resource has this property with this value"

- Validation tests

Help you "fail fast" by making sure your AWS CDK constructs raise errors when you pass them invalid data

Let's build!



Getting started with this workshop

- As a participant, you will have access to an AWS account with any optional pre-provisioned infrastructure and IAM policies needed to complete this workshop
- The AWS account will only be available for the duration of this workshop; you will lose access to the account thereafter
- The optional pre-provisioned infrastructure will be deployed to a specific region; check your workshop content to determine whether other Regions will be used
- Be sure to review the terms and conditions of the event; do not upload any personal or confidential information in the account

Step 1: Sign in via your preferred method

<https://s12d.com/dop401>



aws workshop studio

Workshop Studio > Sign in

Sign in

Choose a preferred sign-in method

Email one-time password (OTP)

Enter your personal or corporate email to receive a one-time password

Login with Amazon

Login with your Amazon.com retail account

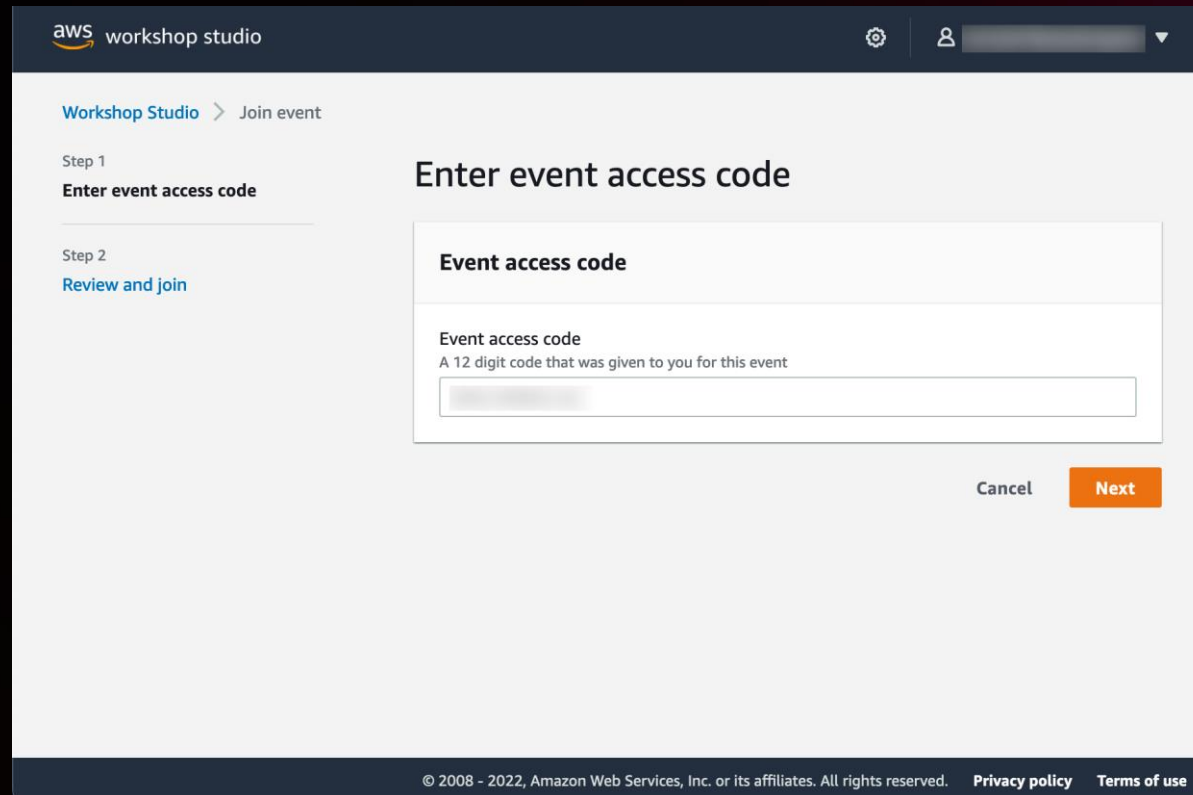
Amazon employee

Login with your Amazon Corporate account. Only for Amazon Employees.

© 2008 - 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved. [Privacy policy](#) [Terms of use](#)

Step 2: Enter event access code

Enter 12-digit event access code if needed



The screenshot shows the AWS Workshop Studio interface. At the top, there's a header with the AWS logo and 'workshop studio' text. Below this, a breadcrumb trail shows 'Workshop Studio > Join event'. The main content area is titled 'Enter event access code'. On the left, a sidebar indicates 'Step 1: Enter event access code' (current step) and 'Step 2: Review and join'. The main form area has a title 'Event access code' and a description: 'Event access code: A 12 digit code that was given to you for this event'. Below this is a text input field. At the bottom right of the form, there are 'Cancel' and 'Next' buttons. The footer contains copyright information: '© 2008 - 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.' and links to 'Privacy policy' and 'Terms of use'.

aws workshop studio

Workshop Studio > Join event

Step 1
Enter event access code

Step 2
Review and join

Enter event access code

Event access code

Event access code
A 12 digit code that was given to you for this event

Cancel Next

© 2008 - 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved. [Privacy policy](#) [Terms of use](#)

Step 3: Review terms and join event

aws workshop studio

Workshop Studio > Join event

Step 1
[Enter event access code](#)

Step 2
Review and join

Review and join

Event details

Name	Start time	Duration	Level
AWS General Immersion Day	9/23/2022 01:13 AM	12 hours	-

Description

AWS General Immersion Day

Terms and Conditions

Read and accept before joining the event

1. By using AWS Workshop Studio for the relevant event, you agree to the AWS Event Terms and Conditions and the AWS Acceptable Use Policy. You acknowledge and agree that are using an AWS-owned account that you can only access for the duration of the relevant event. If you find residual resources or materials in the AWS-owned account, you will make us aware and cease use of the account. AWS reserves the right to terminate the account and delete the contents at any time.
2. You will not: (a) process or run any operation on any data other than test data sets or lab-approved materials by AWS, and (b) copy, import, export or otherwise create derivate works of materials provided by AWS, including but not limited to, data sets.
3. AWS is under no obligation to enable the transmission of your materials through Event Engine and may, in its discretion, edit, block, refuse to post, or remove your materials at any time.
4. Your use of AWS Workshop Studio will comply with these terms and all applicable laws, and your access to AWS Workshop Studio will immediately and automatically terminate if you do not comply with any of these terms or conditions.

☒ I agree with the Terms and Conditions

Cancel

Previous

Join event

© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Thank you!



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

