



Dublin Cloud Day

DUBLIN | MAR 28TH 2023



404

Presentation not found

Did someone hit the wrong button? Did you know about this? Can you go back?

DCD-T7

Infrastructure-as-Code

Gerrard Cowburn (he/him)

Senior Solutions Architect
Amazon Web Services

Richard Maher

Platform Architect
FINEOS



Agenda

Intro to Infrastructure-as-Code (IaC)

AWS CloudFormation vs AWS Cloud Development Kit (CDK)

CDK Concepts

CDK Constructs Deep Dive

FINEOS CDK Journey



Reliably and consistently
**provisioning and configuring
infrastructure is foundational
for DevOps** and fast
software delivery

Manual infrastructure processes
can lack **consistency**, a **single
source of truth**, and **reliable
detection/remediation** of
provisioning errors



Infrastructure as code allows organizations to automate and manage resources consistently



Use version-controlled repositories to create single source of truth



Roll back changes to a previous version as needed



Share and enforce best practices more consistently

AWS has resources for infrastructure as code



AWS CloudFormation

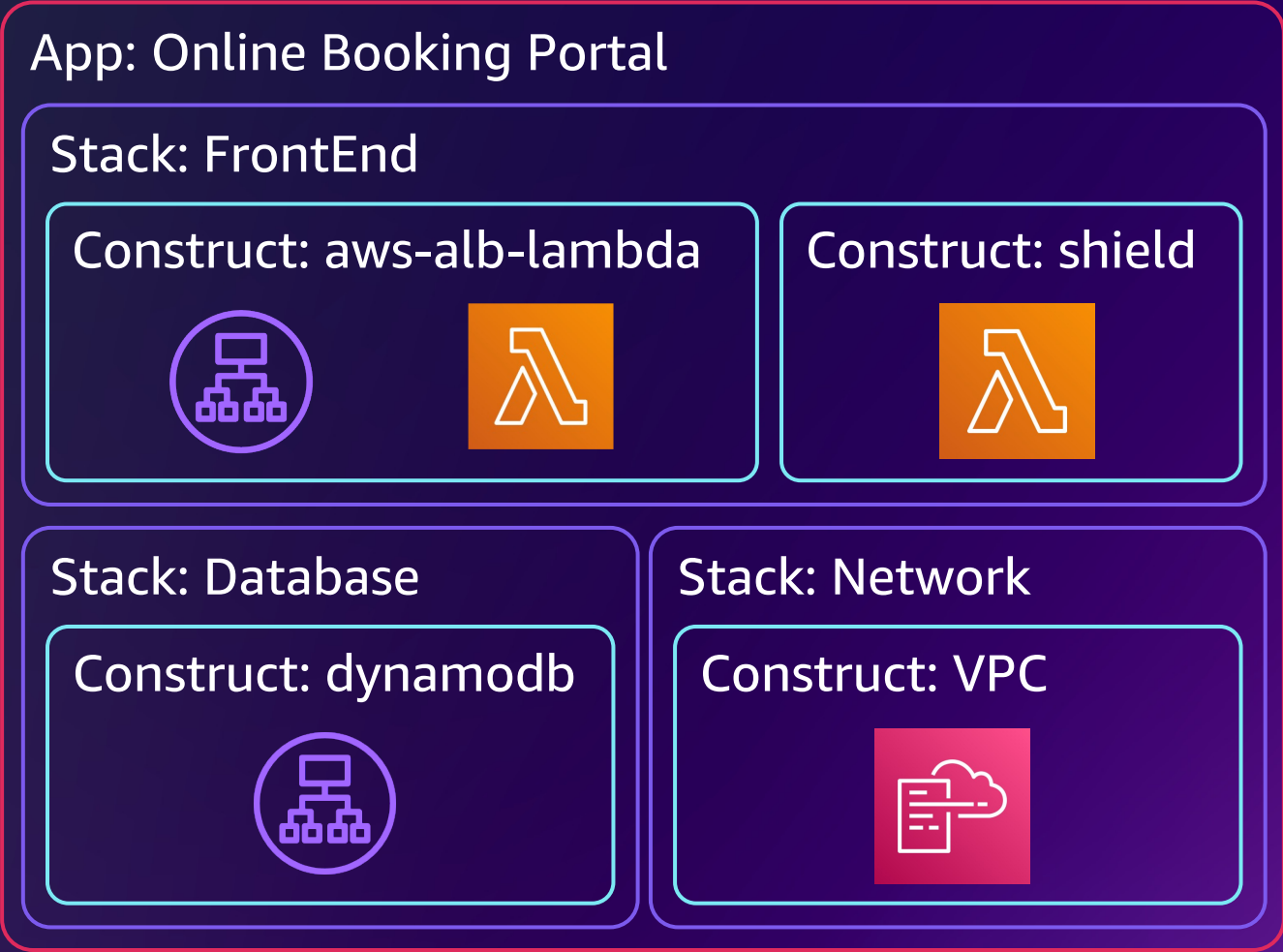
- ✓ Define templates with YAML or JSON
- ✓ Provision templates quickly and consistently
- ✓ Manage templates throughout lifecycle



AWS CDK

- ✓ Use Python, Java, .NET, TypeScript, or Go
- ✓ Class libraries of constructs with sensible defaults
- ✓ Compose and share custom constructs

CDK concepts



Construct levels

L3+

Purpose-built constructs

Opinionated abstractions

L2

AWS Constructs

High level service constructs

L1

CloudFormation Resources

Automatically generated

L2 construct sample

```
new Bucket(scope: Construct, id: string, props?: BucketProps);
```

```
l2Bucket = new s3.Bucket(this, "my-first-cdk-bucket", optionalBucketProps);
```

```
const optionalBucketProps: s3.BucketProps = {  
  blockPublicAccess: s3.BlockPublicAccess.BLOCK_ALL,  
  bucketKeyEnabled: true,  
  encryption: s3.BucketEncryption.KMS,  
  enforceSSL: true,  
  ...  
}
```



L3 construct simple sample

```
export class SimpleCloudFrontS3 extends Construct {  
  constructor(scope: Construct, id: string, props: CloudfrontS3Props) {  
    super(scope, id);  
  }  
}
```

```
}  
}
```

L3 construct simple sample

```
export class SimpleCloudFrontS3 extends Construct {  
  constructor(scope: Construct, id: string, props: CloudfrontS3Props) {  
    super(scope, id);  
  
    const s3Bucket = new s3.Bucket(this, props.bucketName);  
  
  }  
}
```

```
}  
}
```

L3 construct simple sample

```
export class SimpleCloudFrontS3 extends Construct {
  constructor(scope: Construct, id: string, props: CloudfrontS3Props) {
    super(scope, id);

    const s3Bucket = new s3.Bucket(this, props.bucketName);

    const defaultDistributionProps: cf.DistributionProps = {
      defaultBehavior: {
        origin: new S3Origin(this.s3Bucket)
      }
    }
  }
}
```

L3 construct simple sample

```
export class SimpleCloudFrontS3 extends Construct {
  constructor(scope: Construct, id: string, props: CloudfrontS3Props) {
    super(scope, id);

    const s3Bucket = new s3.Bucket(this, props.bucketName);

    const defaultDistributionProps: cf.DistributionProps = {
      defaultBehavior: {
        origin: new S3Origin(this.s3Bucket)
      }
    }
    const cloudFrontWebDistribution = new cf.Distribution(this,
      props.distributionName, defaultDistributionProps);
  }
}
```

L3 construct advanced sample

```
export class MyCloudFrontS3 extends Construct {  
  constructor(scope: Construct, id: string, props: CloudfrontS3Props) {  
    super(scope, id);  
  }  
}
```



L3 construct advanced sample

```
export class MyCloudFrontS3 extends Construct {  
  constructor(scope: Construct, id: string, props: CloudfrontS3Props) {  
    super(scope, id);  
  
    if (props.bucketProps) {  
      validateProvidedBucketProps(props.bucketProps);  
    }  
  }  
}
```



L3 construct advanced sample

```
export class MyCloudFrontS3 extends Construct {
  constructor(
    super(scope, props) {
    if (props.validate) {
      function validateProvidedBucketProps(bucketProps: OptionalBucketProps) {
        Object.keys(bucketProps).forEach(key => {
          if (!allowedS3BucketProps.has(key)) {
            throw new Error(`Setting key ${key} not permitted`);
          }
        });
      }
    }
  }
}
```



L3 construct advanced sample

```
export class MyCloudFrontS3 extends Construct {
  constructor(scope: Construct, id: string, props: CloudfrontS3Props) {
    super(scope, id);

    if (props.bucketProps) {
      validateProvidedBucketProps(props.bucketProps);
    }
    const defaultBucketProps: s3.BucketProps = { ... } // Set default bucket props
```



L3 construct advanced sample

```
export class MyCloudFrontS3 extends Construct {
  constructor(scope: Construct, id: string, props: CloudfrontS3Props) {
    super(scope, id);

    if (props.bucketProps) {
      validateProvidedBucketP
    }
    const defaultBucketProps:
  }

  const defaultBucketProps: s3.BucketProps = {
    blockPublicAccess: s3.BlockPublicAccess.BLOCK_ALL,
    bucketKeyEnabled: true,
    encryption: s3.BucketEncryption.KMS,
    enforceSSL: true
  }
```



L3 construct advanced sample

```
export class MyCloudFrontS3 extends Construct {
  constructor(scope: Construct, id: string, props: CloudfrontS3Props) {
    super(scope, id);

    if (props.bucketProps) {
      validateProvidedBucketProps(props.bucketProps);
    }
    const defaultBucketProps: s3.BucketProps = { ... } // Set default bucket props
    const bucketProps: s3.BucketProps = { // Merge input with defaults
      ...props.bucketProps,
      ...defaultBucketProps
    }
  }
}
```



L3 construct advanced sample

```
export class MyCloudFrontS3 extends Construct {
  constructor(scope: Construct, id: string, props: CloudfrontS3Props) {
    super(scope, id);

    if (props.bucketProps) {
      validateProvidedBucketProps(props.bucketProps);
    }
    const defaultBucketProps: s3.BucketProps = { ... } // Set default bucket props
    const bucketProps: s3.BucketProps = { // Merge input with defaults
      ...props.bucketProps,
      ...defaultBucketProps
    }
    this.s3Bucket = new s3.Bucket(this, props.bucketName, bucketProps);
  }
}
```



L3 construct advanced sample

```
export class MyCloudFrontS3 extends Construct {
  constructor(scope: Construct, id: string, props: CloudfrontS3Props) {
    super(scope, id);

    if (props.bucketProps) {
      validateProvidedBucketProps(props.bucketProps);
    }
    const defaultBucketProps: s3.BucketProps = { ... } // Set default bucket props
    const bucketProps: s3.BucketProps = { // Merge input with defaults
      ...props.bucketProps,
      ...defaultBucketProps
    }
    this.s3Bucket = new s3.Bucket(this, props.bucketName, bucketProps);

    // Remaining CloudFront Code
  }
}
```



Construct Hub



Construct Hub

Getting Started | Documentation | Contribute

Simplify cloud

Find and use open source

Search 900+ constructs

One home for all CDKs

Find libraries for AWS Cloud Development Kit (AWS CDK), which generates AWS CloudFormation templates, CDK for Terraform (CDKtf), which generates HashiCorp Terraform configuration files, and CDK for Kubernetes (CDK8s), which generates Kubernetes manifests.

Support

Define, build, and deploy infrastructure programs using TypeScript

Find documentation and code examples for your application

CloudFrontToS3

Initializers

```
import { CloudFrontToS3 } from '@aws-solutions-constructs/aws-cloudfront-s3'  
  
new CloudFrontToS3(scope: Construct, id: string, props: CloudFrontToS3Props)
```

NAME	TYPE	DESCRIPTION
<code>scope</code>	<code>Construct</code>	- represents the scope for all the resources.
<code>id</code>	<code>string</code>	- this is a a scope-unique id.
<code>props</code>	<code>CloudFrontToS3Props</code>	- user provided props for the construct.



Construct Hub – sample extension

```
export class SolutionBasedCloudFrontS3 extends CloudFrontToS3 {  
  constructor(scope: Construct, id: string, props: CloudfrontS3Props) {  
  
    if (props.bucketProps) {  
      // Do input validation  
    }  
  
    const defaultBucketProps: s3.BucketProps = { ... } // Set default bucket props
```



Construct Hub – sample extension

```
export class SolutionBasedCloudFrontS3 extends CloudFrontToS3 {
  constructor(scope: Construct, id: string, props: CloudfrontS3Props) {

    if (props.bucketProps) {
      // Do input validation
    }

    const defaultBucketProps: s3.BucketProps = { ... } // Set default bucket props
    const cloudFrontToS3Props: CloudFrontToS3Props = {
      bucketProps: { // Merge bucket props input with defaults
        ...props.bucketProps,
        ...defaultBucketProps
      },
      insertHttpSecurityHeaders: true // Set required CloudFront HTTP Security Headers
    }
  }
}
```



Construct Hub – sample extension

```
export class SolutionBasedCloudFrontS3 extends CloudFrontToS3 {
  constructor(scope: Construct, id: string, props: CloudfrontS3Props) {

    if (props.bucketProps) {
      // Do input validation
    }
    const defaultBucketProps: s3.BucketProps = { ... } // Set default bucket props
    const cloudFrontToS3Props: CloudFrontToS3Props = {
      bucketProps: { // Merge bucket props input with defaults
        ...props.bucketProps,
        ...defaultBucketProps
      },
      insertHttpSecurityHeaders: true // Set required CloudFront HTTP Security Headers
    }
    super(scope, id, cloudFrontToS3Props); // Invoke original Construct Hub construct
  }
}
```



FINEOS



Richard Maher (Platform Architect)

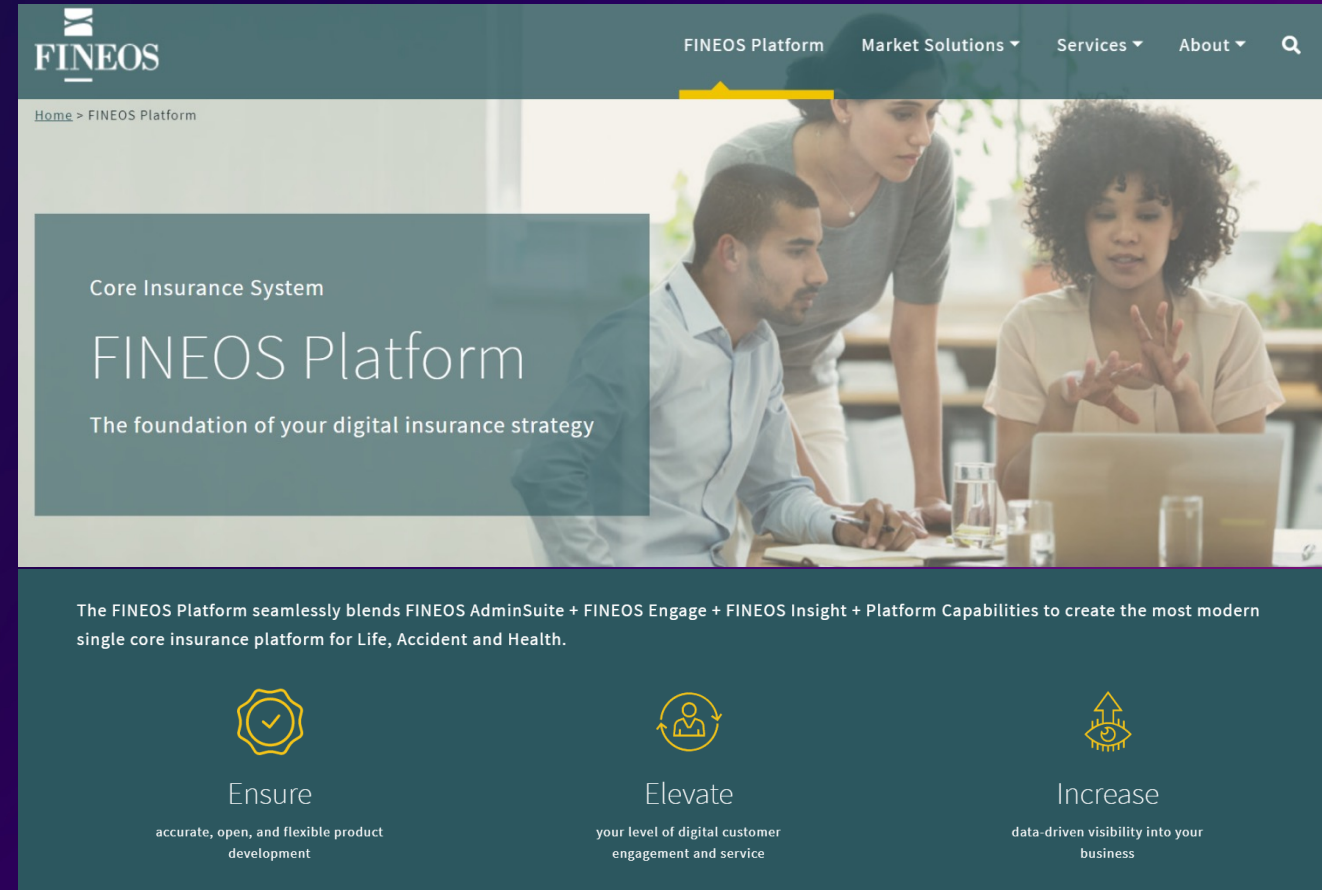
Joined FINEOS in 2007

FINEOS

FINEOS is the global market leader among Life, Accident and Health insurance technology providers

FINEOS serves over 50 customers globally

FINEOS serves customers from 8 locations around the world



Where we started



From the outset we aimed for no manual changes in production

Relatively small AWS services and account footprint

A single team of IaC developers co-located

AWS CDK not available at this time



What happened next



Increasing complexity of our cloud service offering with increased AWS account footprint

Pervasive need across the organisation for teams to develop and build cloud solutions

We were starting to see a monolith...

- Increase in single codebase
- Increase in deployment time
- Increase in number of deployment time failures



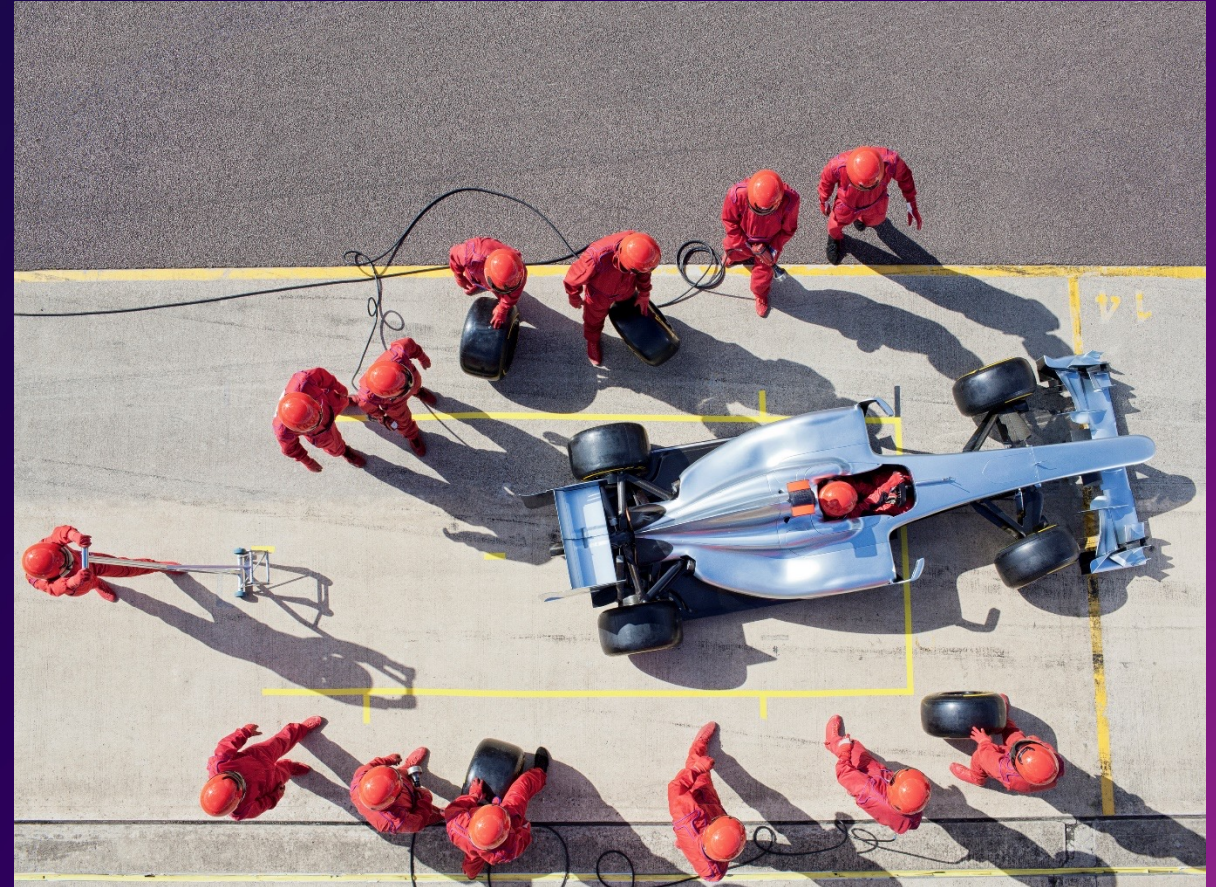
How CDK is solving our problems



Through modern software development practices, increase:

- quality
- re-use
- standardisation
- modularisation

- ⇒ Unit testing
- ⇒ Mock integration
- ⇒ Debug
- ⇒ Synth / compile
- ⇒ Shared libraries, functions and constructs



Where we are now



30+ CDK apps for production cloud and cloud tool chain with ownership across 10 teams

CDK is our de-facto standard IAC tool

All engineering groups are developing with CDK

- Upskilled on software engineering practices, AWS and CloudFormation

Configuration management construct

Developed build, release & orchestration pipelines

Where we're going



Continue to componentise create shared services and use CDK

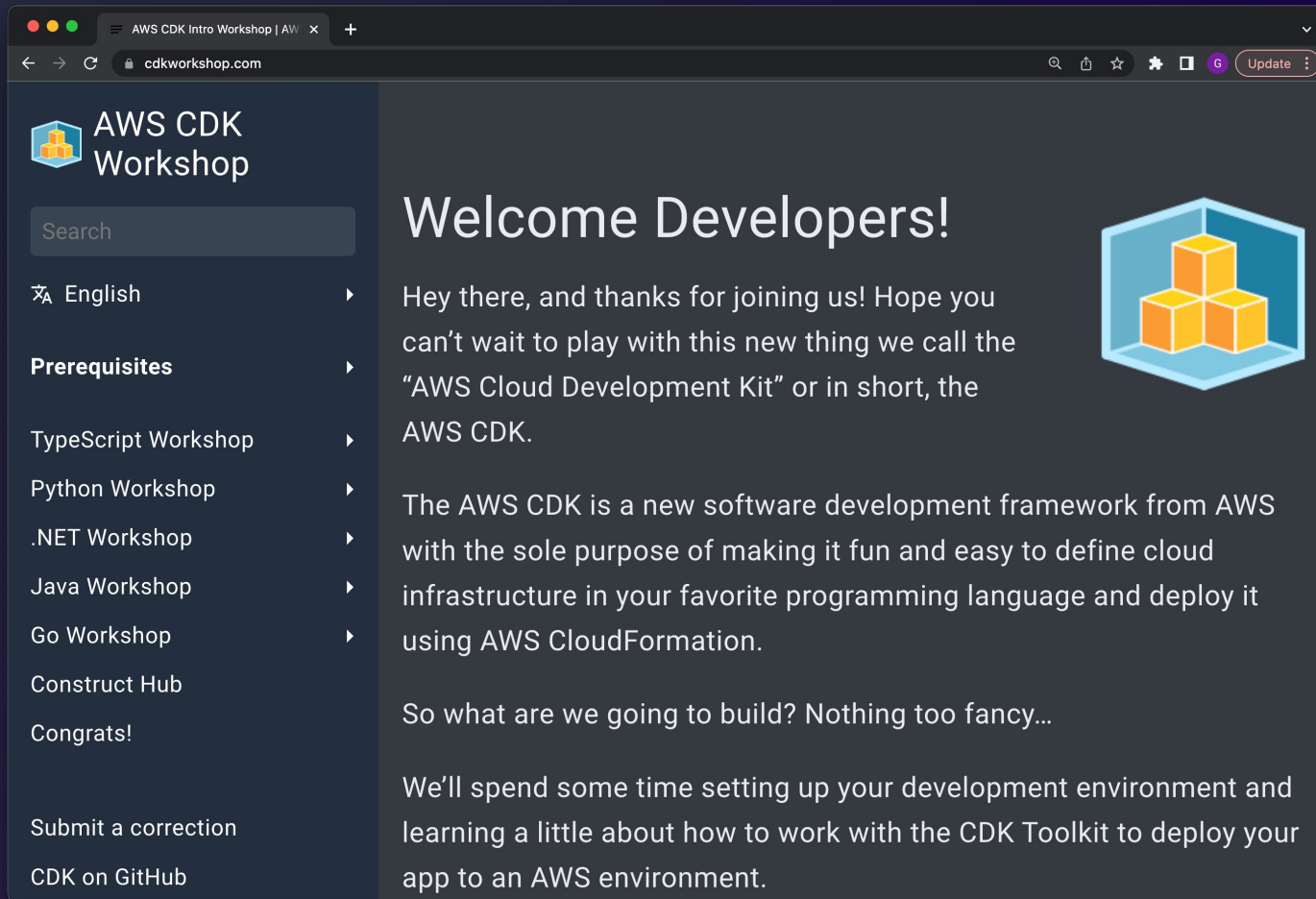
We plan to use CDK import to migrate from older tech stacks

Multi app, multi account deployment orchestration



Give it a try!

Get hands on! cdkworkshop.com

A screenshot of a web browser displaying the AWS CDK Workshop website. The browser's address bar shows 'cdkworkshop.com'. The page has a dark theme. On the left is a navigation sidebar with a search bar, a language selector set to 'English', and a 'Prerequisites' section listing 'TypeScript Workshop', 'Python Workshop', '.NET Workshop', 'Java Workshop', and 'Go Workshop'. Below that are links for 'Construct Hub', 'Congrats!', 'Submit a correction', and 'CDK on GitHub'. The main content area features a large heading 'Welcome Developers!' next to a 3D cube icon. Below the heading is a paragraph: 'Hey there, and thanks for joining us! Hope you can't wait to play with this new thing we call the "AWS Cloud Development Kit" or in short, the AWS CDK.' This is followed by another paragraph: 'The AWS CDK is a new software development framework from AWS with the sole purpose of making it fun and easy to define cloud infrastructure in your favorite programming language and deploy it using AWS CloudFormation.' At the bottom of the main content, it says 'So what are we going to build? Nothing too fancy...' and 'We'll spend some time setting up your development environment and learning a little about how to work with the CDK Toolkit to deploy your app to an AWS environment.'

Further reading...

- aws.amazon.com/cdk
- aws-samples/aws-cdk-examples
- constructs.dev
- cdkpatterns.com
- cdk.dev

Contribute!

- aws/aws-cdk
- aws/jsii



Thank you!

Gerrard Cowburn

[linkedin.com/in/gerrardcowburn](https://www.linkedin.com/in/gerrardcowburn)

Richard Maher

[linkedin.com/in/richard-maher-b990962/](https://www.linkedin.com/in/richard-maher-b990962/)





Please complete
the session survey