#### CDK Day May 2022 - Track 2















#### 3,026 views Streamed live on May 26, 2022

Cloud Development Kit (CDK) is a developer tool built on the open source Constructs model. CDK Day is a one-day conference that attempts to showcase the brightest and best of CDK from AWS CDK, CDK for Terraform (CDKtf), CDK for Kubernetes (cdk8s), and projen. Let's talk serverless, Kubernetes and multi-cloud all on the same day.

This is the livestream for Track 2 of CDK Day, featuring talks from Ansgar Mertens, Jenna Krolick, Bill Penberthy, and more. For the full agenda, please see https://www.cdkday.com/.

For the livestream of talks from Track 1, please head over to https://bit.ly/cdkday2022-track1 For the livestream of talks from Track 3, please head over to https://bit.ly/cdkday2022-track3

00:00 Holding Page

38:39 Introductions

42:10 Building security with CDK

1:05:12 AWS Adapter: Using AWS CDK constructs with the CDK for Terraform

1:27:32 Using-Driver Composable Infrastructure with CDK

1:50:31 CDK & Team Topologies: Enabling the Optimal Platform Team

2:05:02 Build Event Driver Architectures with the AWS CDK

2:25:40 Snapshot Testing and CDK

2:50:02 Discussion: The local cloud - ideas to ensure developer productivity in serverless architectures

3:26:03 Schema-Driver OpenAPI Development with AWS CDK

3:47:52 Selling CDK to an Old-School DevOps org

4:17:12 Simplifying data pipelines by sharing AWS CDK Constructs within the team

4:36:44 Improved IAM through CDK

4:53:48 Closing comments

# https://www.cdkday.com/coc/

y-track1 #cdkday-track2 #cdkday-track3

cdk.dev #cdk



# **AWS Adapter**

Using AWS CDK constructs with the CDK for Terraform

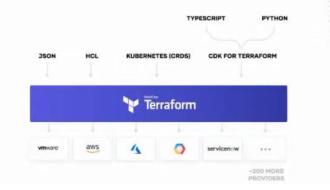
### **CDK for Terraform**



Use Terraform with programming constructs in high-level languages

### Supported Languages

- TypeScript / JavaScript
- Java
- Python
- C#
- Go



## Example

#### HCL

### **TypeScript**

```
new Instance(this, 'web', {
   ami: 'ami-0848da720bb07de35',
   instanceType: 't3.micro',
   tags: {
      Name: 'HelloWorld'
   }
});
```

## A year ago ...

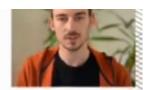




April 2021 - Extending constructs of the CDK for Terraform, Ansgar Mertens

## **AWS CDK**

Layer 2 constructs



```
const lambda = new lambda.Function(this, 'Lambda', { ... });
const bucket = new Bucket(this, 'MyBucket');
bucket.grantReadWrite(lambda);
```

## **AWS Adapter**

### Use AWS CDK constructs with CDKTF



```
// import adapter and AWS CDK constructs
import { aws_lambda } from "aws-cdk-lib";
import { AwsTerraformAdapter } from "@cdktf/aws-cdk";

// instantiate adapter, passing a TerraformStack ("this")
const awsAdapter = new AwsTerraformAdapter(this, "adapter");
// pass the adapter to AWS CDK constructs as scope when using them
const lambdaFn = new aws_lambda.Function(awsAdapter, "lambda", { . . . });
```

## How does it work?

## **Construct Output**

CDKTF vs. AWS CDK



CDKTF constructs

Terraform config

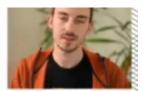
AWS CDK constructs CloudFormation config

## CfnElement

```
185
   106
           * Returns the CloudFormation 'snippet' for this entity. The snippet will only be merged
   107
           * at the root level to ensure there are no identity conflicts.
   109
           * For example, a Resource class will return something like:
   110
           * Resources: {
   111
               [this.logicalId]: {
   112
   113
                 Type: this.resourceType,
                 Properties: this.props,
   114
   115
                 Condition: this condition
   117
           * }
   118
   119
   128
           * @internal
   121
*** 122
         public abstract _toCloudFormation(): object;
```



## **TerraformElement**



```
43

••• 44 public toTerraform(): any {

45 return {};

46 }

47
```

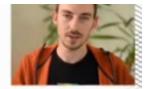
## Construct



```
462 */
... 463 constructor(scope: Construct, id: string) {
464 this.node = new Node(this, scope, id);
465
```

## Highlight

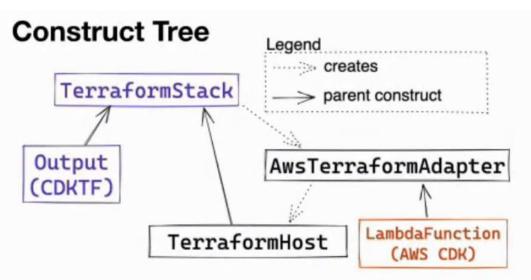
### Passing a different scope to AWS CDK constructs



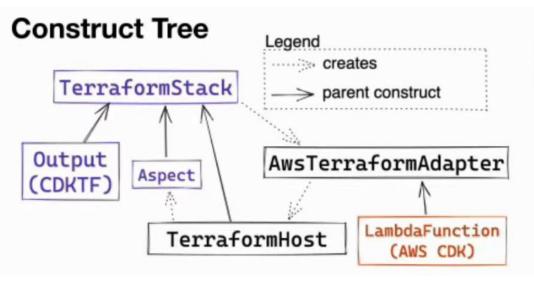
```
// import adapter and AWS CDK constructs
import { aws_lambda } from "aws-cdk-lib";
import { AwsTerraformAdapter } from "@cdktf/aws-cdk";

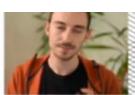
// instantiate adapter, passing a TerraformStack ("this")
const awsAdapter = new AwsTerraformAdapter(this, "adapter");

// pass the adapter to AWS CDK constructs as scope when using them
const lambdaFn = new aws_lambda.Function(awsAdapter, "lambda", { . . . });
```





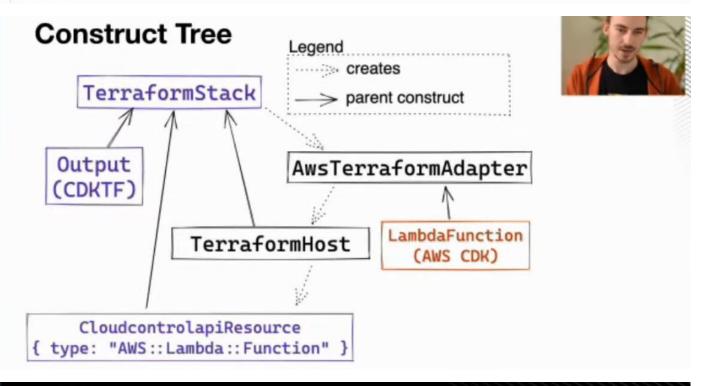




## Aspects?



```
37
    export class AwsTerraformAdapter extends Stack {
38
      constructor(scope: Construct, id: string) {
39
        super(undefined, id);
41
       const host = new TerraformHost(scope, id, this);
43
      Aspects.of(scope).add({
         visit: (node) => {
45
           if (node === scope) {
              // TODO: invokeAWSAspects(this); -> find usages of AWSAspects in AWS constructs
              host.convert();
49
          },
50
        });
51
52
```



# Let's look at the code!

```
EXPLORER
                                                                                                         Ш ...
                         TS aws-adapter ts X
CDKTF-AWS-CDK [GIT...
                          src > TS aws-adapter.ts > & AwsTerraformAdapter
                           36
  > scripts
                                 export class AwsTerraformAdapter extends Stack {
                           37
  v src
                           38
                                   constructor(scope: Construct, id: string) {
   ) aws
                           39
                                     super(undefined, id);
   > awscc
                           40
   > mapping
                           41
                                     const host = new TerraformHost(scope, id, this);
                           42
   > tests
                                     Aspects.of(scope).add({
                           43
   > time
                           44
                                       visit: (node) => {
  TS aws-adapter.ts
                           45
                                         if (node === scope) {
  TS cfn.ts
                           46
                                           // TODO: invokeAWSAspects(this); -> find usages of AWSAspects in A
  TS es2019.ts
                           47
                           48
  TS index.ts
                           49
                                       1.
  TS type-utils.ts
                           50
                                     });
 eslintrc.json
                           51
 .gitattributes
                           52
                           53

    gitignore

                                 class TerraformHost extends Construct {
                           54
> OUTLINE
                           55
                                   private awsPartition?: datasources.DataAwsPartition;
                           56
                                 private awsRegion?: datasources.DataAwsRegion;
> TIMELINE
  EXPLORER
                         TS aws-adapter.ts X
                                                                                                         □ ...

∨ CDKTF-AWS-CDK [GIT...

                          src > TS aws-adapter.ts > 😘 AwsTerraformAdapter > 🛇 constructor > 🛇 visit
  > scripts
                           54
                                 class TerraformHost extends Construct {
                           55
                                   private awsPartition?: datasources.DataAwsPartition;
 V SEC
                                   private awsRegion?: datasources.DataAwsRegion;
                           56
  ) aws
                                   private awsCallerIdentity?: datasources.DataAwsCallerIdentity;
                           57
   > awscc
                           58
                                   private awsAvailabilityZones: {
   > mapping
                           59
                                   [region: string]: datasources.DataAwsAvailabilityZones;
                           60
                                   } = {};
   ) tests
                           61
                                   private regionalAwsProviders: { [region: string]: AwsProvider ) = {};
   > time
                           62
  TS aws-adapter.ts
                           63
                                   // TODO: expose this via some method?
  TS cfn.ts
                           64
                                   private readonly mappingForLogicalId: {
  TS es2019.ts
                           65
                                     [logicalId: string]: {
                           66
                                       resourceType: string:
  TS index.ts
                           67
                                       mapping: Mapping<TerraformResource>;
  TS type-utils.ts
                                     }:
                           68
 eslintrc.json
                           69
                                   } = {};
 .gitattributes
                           70

    gitignore

                           71
                                   constructor(
                           72
                                     scope: Construct,
OUTLINE
                           73
                                     id: string,
> TIMELINE
                           74
                                     private readonly host: AwsTerraformAdapter
                                                                                                         □ ...
  EXPLORER
                         TS aws-adapter.ts X
                          src > TS aws-adapter.ts > 13 TerraformHost
V CDKTF-AWS-CDK [GIT...
                                     (region: string): datasources.DataAwsAvailaDilityZones;
                           59
  > scripts
                           60
                                   } = {};
  v src
                           61
                                   private regionalAwsProviders: { [region: string]: AwsProvider } = {};
                           62
   > aws
                                   // TODO: expose this via some method?
                           63
   ) awscc
                           64
                                   private readonly mappingForLogicalId: {
   > mapping
                           65
                                     [logicalId: string]: {
   > tests
                           66
                                       resourceType: string;
   > time
                           67
                                       mapping: Mapping<TerraformResource>;
                           68
                                     };
  TS aws-adapter.ts
                           69
                                   } = {};
  TS cfn.ts
                           70
  TS es2019.ts
                           71
                                   constructor(
  TS index.ts
                           72
                                     scope: Construct,
                           73
                                     id: string,
  TS type-utils.ts
                           74
                                     private readonly host: AwsTerraformAdapter
 @ .eslintrc.json
                                   ) {
                           75
 .gitattributes
                           76
                                   super(scope, id);
 .gitignore
                           77
                           78
> OUTLINE
                           79
                                   convert() {
```

```
TS aws-adapter.ts X
src > TS aws-adapter.ts > % TerraformHost > 分 convert
 18
 79
         convert() {
 80
           for (const r of this.host.node.findAll()) {
 81
             if (r instanceof CfnElement) {
 82
               const cfn = this.host.resolve(
 83
                 (r as any)._toCloudFormation()
               ) as CloudFormationTemplate;
 84
 85
               for (const [logical, value] of Object.entries(cfn.Resources || {}})) {
 86
               this.newTerraformResource(this, logical, value);
 87
 88
               for (const [conditionId, condition] of Object.entries(
 89
                cfn.Conditions || {}
               1) {
 90
 91
               this.newTerraformLocalFromCondition(this, conditionId, condition);
 92
 93
               for (const [outputId, args] of Object.entries(cfn.Outputs || {})) {
 94
               this.newTerraformOutput(this, outputId, args);
 95
 96
 97
 98
                                                                                                     □ ...
TS aws-adapter.ts X
src > TS aws-adapter.ts > 1 TerraformHost > 1 newTerraformResource
137
138
         private hewTerraformResource(
139
           scope: Construct,
140
           logicalId: string,
141
           resource: CloudFormationResource
142
         ): TerraformResource | null {
143
           // TODO: add debug log console.log(JSON.stringify(resource, null, 2));
144
           const m = findMapping(resource.Type);
145
           if (!m) {
146
            throw new Error('no mapping for ${resource.Type}');
147
148
149
           const props = this.processIntrinsics(resource.Properties ?? {});
150
           const conditionId = resource.Condition;
151
152
           this.mappingForLogicalId[logicalId] = {
153
             resourceType: resource.Type,
154
             mapping: m.
155
           };
156
157
           const res = m.resource(scope, logicalId, props);
TS aws-adapter.ts X
src > TS aws-adapter.ts > % TerraformHost > 分 newTerraformResource
154
           mapping: m,
155
           }:
156
           const res = m.resource(scope, logicalId, props);
157
158
159
           if (conditionId) {
160
             if (!res) {
161
               throw new Error(
162
                 'Condition has been found on resource that has no representation in Terraform: ${resource.
163
               );
                                     I
164
165
166
             res.count = Token.asNumber(
              conditional(this.getConditionTerraformLocal(conditionId), 1, 0)
167
168
169
170
           const keys = Object.keys(props).filter((k) => props[k] !== undefined);
171
172
           if (keys.length > 0) {
173
             throw new Error(
174
               `cannot map some properties of ${resource.Type}: ${JSON.stringify(
```

```
TS aws-adapter.ts X
src > TS aws-adapter.ts > % TerraformHost > ♦ newTerraformResource
              conditional(this.getConditionTerraformLocal(conditionId), 1, 0)
168
169
170
           const keys = Object.keys(props).filter((k) => props[k] !== undefined);
171
172
           if (keys.length > 0) {
173
             throw new Error(
174
               'cannot map some properties of ${resource.Type}: ${JSON.stringify(
175
                props,
176
                undefined,
177
               2
178
              13.
179
             );
180
181
182
           return res;
183
184
185
         private newTerraformOutput(scope: Construct, outputId: string, args: any) {
           return new TerraformOutput(scope, outputId, {
186
187
            value: this.processIntrinsics(args.Value),
```

## Recap



- 1. Second, disjunct construct tree
- 2. Aspect for conversion step
  - Traverse second construct tree
  - Convert constructs
  - Add results to original construct tree
- 3. Original construct tree gets synthesized

### Links

#### Get started with the CDK for Terraform



https://cdk.tf

https://cdk.tf/adapter

https://learn.hashicorp.com/tutorials/terraform/cdktf

https://discuss.hashicorp.com/c/terraform-core/cdk-for-terraform/47