
AWS
re:Invent

DOP309

Test twice, deploy once: Testing infrastructure code on AWS

Darko Meszaros

Developer Advocate
AWS

AWS re:Invent

© 2020, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Talk pipeline

Why infrastructure as code?

Testing AWS CloudFormation

Even more, better testing with AWS CDK



© 2020, Amazon Web Services, Inc. or its affiliates. All rights reserved.

What is infrastructure as code?

Why infrastructure as code?


8562hd90096

Infrastructure as code

From Wikipedia, the free encyclopedia

Overview [edit]

IaC grew as a response to the difficulty posed by utility computing and second-generation web frameworks. In 2006, the launch of Amazon Web Services' Elastic Compute Cloud and the 1.0 version of Ruby on Rails just months before[2] created widespread scaling problems in the enterprise that were previously experienced only at large, multi-national companies.[3] With new tools emerging to handle this ever growing field, the idea of IaC was born. The thought of modelling infrastructure with code, and then having the ability to design, implement, and deploy applications infrastructure with known software best practices appealed to both software developers and IT infrastructure administrators. The ability to treat infrastructure like code and use the same tools as any other software project would allow developers to rapidly deploy applications.[4]

Source: https://en.wikipedia.org/wiki/Infrastructure_as_code

```
import cdk = require('@aws-cdk/core');
import ec2 = require('@aws-cdk/aws-ec2');
import lambda = require('@aws-cdk/aws-lambda');
import iam = require('@aws-cdk/aws-iam');
import elbv2 = require('@aws-cdk/aws-elasticloadbalancingv2');
import elbv2Targets = require('@aws-cdk/aws-elasticloadbalancingv2-targets');


export class AlbGoingGlobalWithServerlessStack extends cdk.Stack {
  constructor(scope: cdk.Construct, id: string, props?: cdk.StackProps) {
    super(scope, id, props);

    // The code that defines your stack goes here

    // VPC
    const vpc = new ec2.Vpc(this, 'VPC');

    // The code that defines your stack goes here
    const getGlobalALB = new lambda.Function(this, 'getGlobalALB', {
      runtime: lambda.Runtime.NODEJS_8_10,
      code: lambda.Code.asset('lambda'),
      handler: 'getStuff.handler',
      environment: {
        'STATUS':'200'
      }
    });

    // IAM Policy
    const lambdaDynamoDbStatement = new iam.PolicyStatement();
    lambdaDynamoDbStatement.addActions('*');
```

Infrastructure is code



✓ Make infrastructure changes repeatable and predictable

✓ Release infrastructure changes using the same tools as code changes

✓ Replicate production in a staging environment to enable continuous testing

Speaking of code







We test it!

“Measure twice, cut once.”

Unknown



© 2020, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Testing AWS CloudFormation



AWS re:Invent

© 2020, Amazon Web Services, Inc. or its affiliates. All rights reserved.

AWS CloudFormation

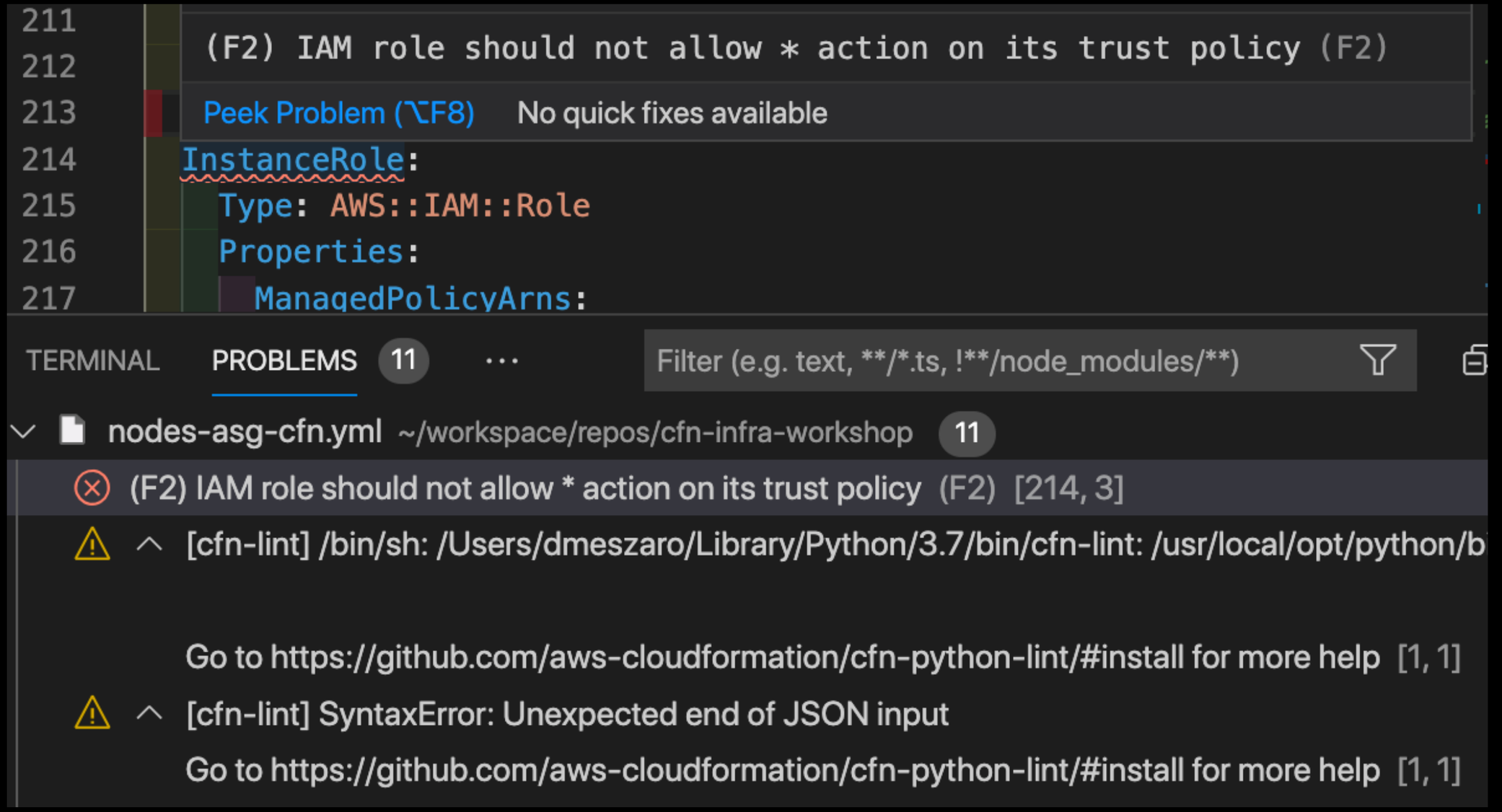
- Infrastructure as code (IaC)
- Provides a common language for you to describe and provision all the infrastructure resources in your cloud environment
- Build and rebuild your infrastructure and applications, without having to perform manual actions or write custom scripts



https://aws.amazon.com/cloudformation/

Where it all begins


tmux /Users/dmeszaro
7 interface EnvStackProps extends cdk.StackProps {
8 prod: boolean;
9 }
10
11 export class LambdaAppconfigStack extends cdk.Stack {
12 constructor(scope: cdk.Construct, id: string, props?: EnvStackProps)
13 super(scope, id, props);
14
15 // Defining the prod or no prod
16 if (props && props.prod) { // prod
17 var apiName = 'appConfigApiPROD'
18 var lambdaVars = {
19 'CONFIG_ENV': 'lambdaProd',
20 'CONFIG_APP': 'lambdaDemo',
21 'CONFIG_CONFIG':'lambdaLimiter',
22 };
23 } else { // not prod
24 var apiName = 'appConfigApiSTAGE'
25 var lambdaVars = {
<RMAL lambda-appconfig-stack.ts unix | utf-8 | typescript 38% 24:1
[8:42:49 PM] Starting compilation in watch mode...
[8:42:53 PM] Found 0 errors. Watching for file changes.
cozy
[I] [0] dmeszaro@a483e7de39da ~/w/r/c/lambda-appconfig (main) →
cozy-session 1 cozy-vim 2020-09-28 20:43 darko@macbook


211
212
(F2) IAM role should not allow * action on its trust policy (F2)
213
Peek Problem (⌥F8)
No quick fixes available
214
InstanceRole:
215
Type: AWS::IAM::Role
216
Properties:
217
ManagedPolicyArns:
TERMINAL
PROBLEMS 11
Filter (e.g. text, **/*.ts, !**/node_modules/**)
nodes-asg-cfn.yml ~/workspace/repos/cfn-infra-workshop 11
(F2) IAM role should not allow * action on its trust policy (F2) [214, 3]
[cfn-lint] /bin/sh: /Users/dmeszaro/Library/Python/3.7/bin/cfn-lint: /usr/local/opt/python/b
Go to https://github.com/aws-cloudformation/cfn-python-lint/#install for more help [1, 1]
[cfn-lint] SyntaxError: Unexpected end of JSON input
Go to https://github.com/aws-cloudformation/cfn-python-lint/#install for more help [1, 1]

The tool behind it: cfn-nag

```
-------------------------------------------------------
| FAIL F2
|
| Resources: ["InstanceRole"]
| Line Numbers: [215]
|
| IAM role should not allow * action on its trust policy
-------------------------------------------------------
| WARN W9
|
| Resources: ["SG", "LBSG"]
| Line Numbers: [82, 106]
|
| Security Groups found with ingress cidr that is not /32
-------------------------------------------------------
| WARN W36
|
| Resources: ["SG", "LBSG"]
| Line Numbers: [82, 106]
|
| Security group rules without a description obscure their purpose and
 the ports and sources/destinations required.

Failures count: 1
Warnings count: 8
```

- Patterns that indicate insecure infrastructure
- Bring your own rules
- Run against multiple templates
- github.com/stelligent/cfn_nag




```
gem install cfn_nag --user
```

Command line linting with cfn-lint



- Code linter for AWS CloudFormation templates
- Can lint against different regions
- Can be configured directly within AWS CloudFormation templates

github.com/aws-cloudformation/cfn-python-lint

```
pip3 install cfn-lint --user
```

Lint twice, deploy once!

```
➜  cfn-infra-workshop git:(master)
➜  cfn-infra-workshop git:(master) cfn-lint nodes-asg-cfn.yml
W3010 Don't hardcode eu-west-1a for AvailabilityZones
nodes-asg-cfn.yml:23:7

W3010 Don't hardcode eu-west-1b for AvailabilityZones
nodes-asg-cfn.yml:34:7

➜  cfn-infra-workshop git:(master) cfn-lint nodes-asg-cfn.yml --regions eu-north-1
W3010 Don't hardcode eu-west-1a for AvailabilityZones
nodes-asg-cfn.yml:23:7

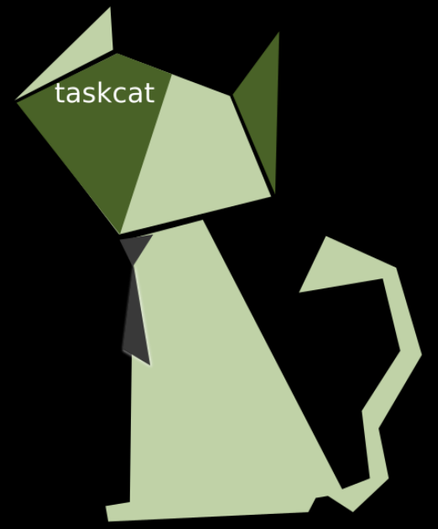
W3010 Don't hardcode eu-west-1b for AvailabilityZones
nodes-asg-cfn.yml:34:7

E3030 You must specify a valid value for InstanceType (t2.micro).
Valid values are ["c5.12xlarge", "c5.18xlarge", "c5.24xlarge", "c5.2xlarge", "c5.4xlarge", "c5.9xlarge", "c5.large", "c5.metal", "c5.xla$
ge", "c5d.12xlarge", "c5d.18xlarge", "c5d.24xlarge", "c5d.2xlarge", "c5d.4xlarge", "c5d.9xlarge", "c5d.large", "c5d.metal", "c5d.xlarge"$
 "d2.2xlarge", "d2.4xlarge", "d2.8xlarge", "d2.xlarge", "g4dn.12xlarge", "g4dn.16xlarge", "g4dn.2xlarge", "g4dn.4xlarge", "g4dn.8xlarge"$
 "g4dn.xlarge", "i3.16xlarge", "i3.2xlarge", "i3.4xlarge", "i3.8xlarge", "i3.large", "i3.metal", "i3.xlarge", "m5.12xlarge", "m5.16xlarg$
", "m5.24xlarge", "m5.2xlarge", "m5.4xlarge", "m5.8xlarge", "m5.large", "m5.metal", "m5.xlarge", "m5d.12xlarge", "m5d.16xlarge", "m5d.24$
large", "m5d.2xlarge", "m5d.4xlarge", "m5d.8xlarge", "m5d.large", "m5d.metal", "m5d.xlarge", "r5.12xlarge", "r5.16xlarge", "r5.24xlarge"$
 "r5.2xlarge", "r5.4xlarge", "r5.8xlarge", "r5.large", "r5.metal", "r5.xlarge", "r5d.12xlarge", "r5d.16xlarge", "r5d.24xlarge", "r5d.2xl$
rge", "r5d.4xlarge", "r5d.8xlarge", "r5d.large", "r5d.metal", "r5d.xlarge", "t3.2xlarge", "t3.large", "t3.medium", "t3.micro", "t3.nano"$
 "t3.small", "t3.xlarge"]
nodes-asg-cfn.yml:153:7

➜  cfn-infra-workshop git:(master)
```

Test at scale with TaskCat

- Tests AWS CloudFormation templates by deploying them
- Uses cfn-lint out of the box for linting
- Deploys to multiple regions with different parameter sets
- Provides report generation and log collection
- Has the ability to build and package lambda functions



github.com/aws-quickstart/taskcat

```
pip3 install taskcat --user
```

TaskCat tests

```yaml
project:
  name: iac-deepdive
  regions:
  - us-west-2
  - eu-north-1
tests:
  us-west-2:
    parameters:
      Key: darko-us-west-2
      InstanceType: t2.micro
      AMI: ami-079f731edfe27c29c
    regions:
      - us-west-2
    template: ./tester-template.yml
  eu-north-1:
    parameters:
      Key: darko-eu-north-1
      InstanceType: t3.micro
      AMI: ami-01a7a49829bda9d79
    regions:
      - eu-north-1
    template: ./tester-template.yml
~
~
1 | .taskcat.yml
```

- Regions
- Blacklisted AZs
- Parameter sets
- Tags

TaskCat test run

```
➜  IaC-DeepDive git:(master) ✗ taskcat test run
 _             _               _
| |_ __ _ ___| | _____ __ _| |_
| __/ _` / __| |/ / __/ _` | __|
| || (_| \__ \   < (_| (_| | |_
 \__\__,_|___/_|\_\___\__,_|\__|



version 0.9.13
[INFO    ] : Lint passed for test us-west-2 on template /Users/dmeszaro/workspace/repos/IaC-DeepDive/tester-template.yml
[INFO    ] : Lint passed for test eu-north-1 on template /Users/dmeszaro/workspace/repos/IaC-DeepDive/tester-template.yml
[S3: -> ] s3://tcat-iac-deepdive-f7tjn6pk/iac-deepdive/tester-template.yml
          ┌ stack Ⓜ tCaT-iac-deepdive-eu-north-1-762869d6831c4a51a13ab38d44037073
          ├ region: eu-north-1
          └ status: CREATE_IN_PROGRESS
          ┌ stack Ⓜ tCaT-iac-deepdive-us-west-2-762869d6831c4a51a13ab38d44037073
          ├ region: us-west-2
          └ status: CREATE_IN_PROGRESS
```

Testing AWS Cloud Development Kit (AWS CDK)



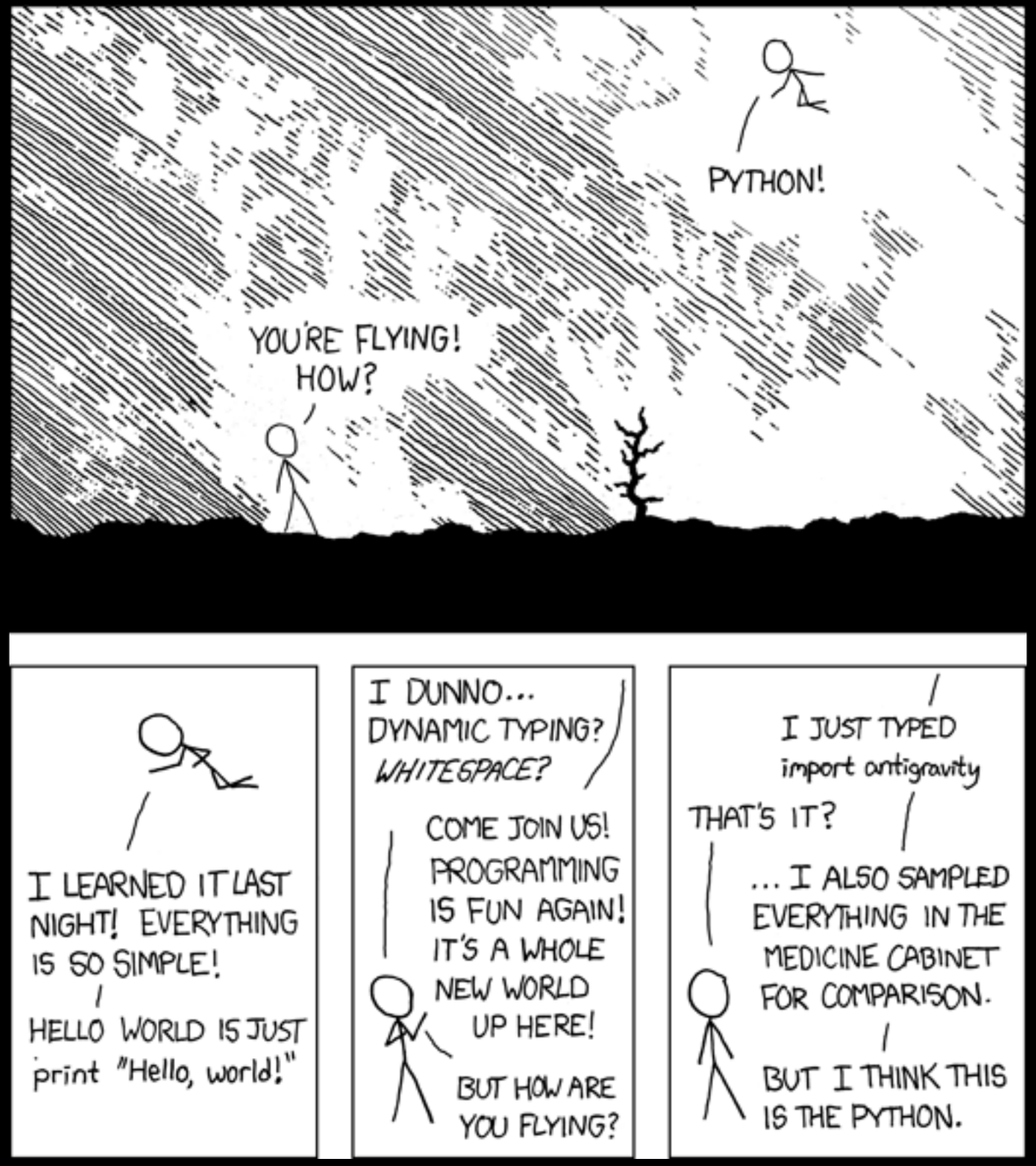
AWS re:Invent

© 2020, Amazon Web Services, Inc. or its affiliates. All rights reserved.



```
{ "json":"true" }

yaml:
  aint:
    - markup
      language
```



Source: https://xkcd.com/353/

Shorten the learning curve with AWS CDK

Brings cloud infrastructure to developers in ways they can understand

- Build cloud infrastructure with the languages they already know
- Use their existing tools and workflows
- Helpful abstractions that remove the need to learn the details
- Vibrant and fast-growing community of developers










How do we do testing with AWS CDK?

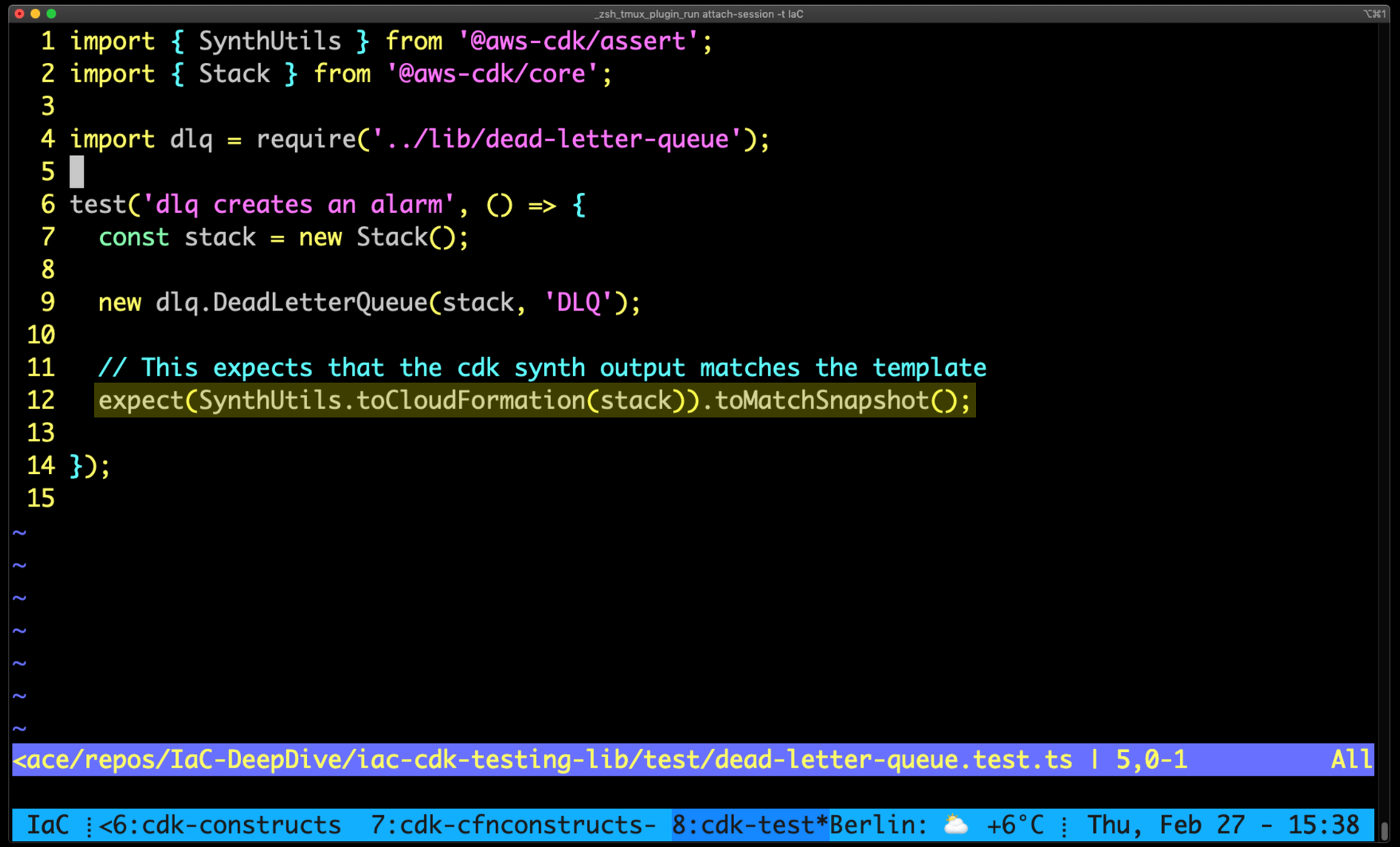
- Snapshot tests
- Fine-grained assertions
- Validation tests



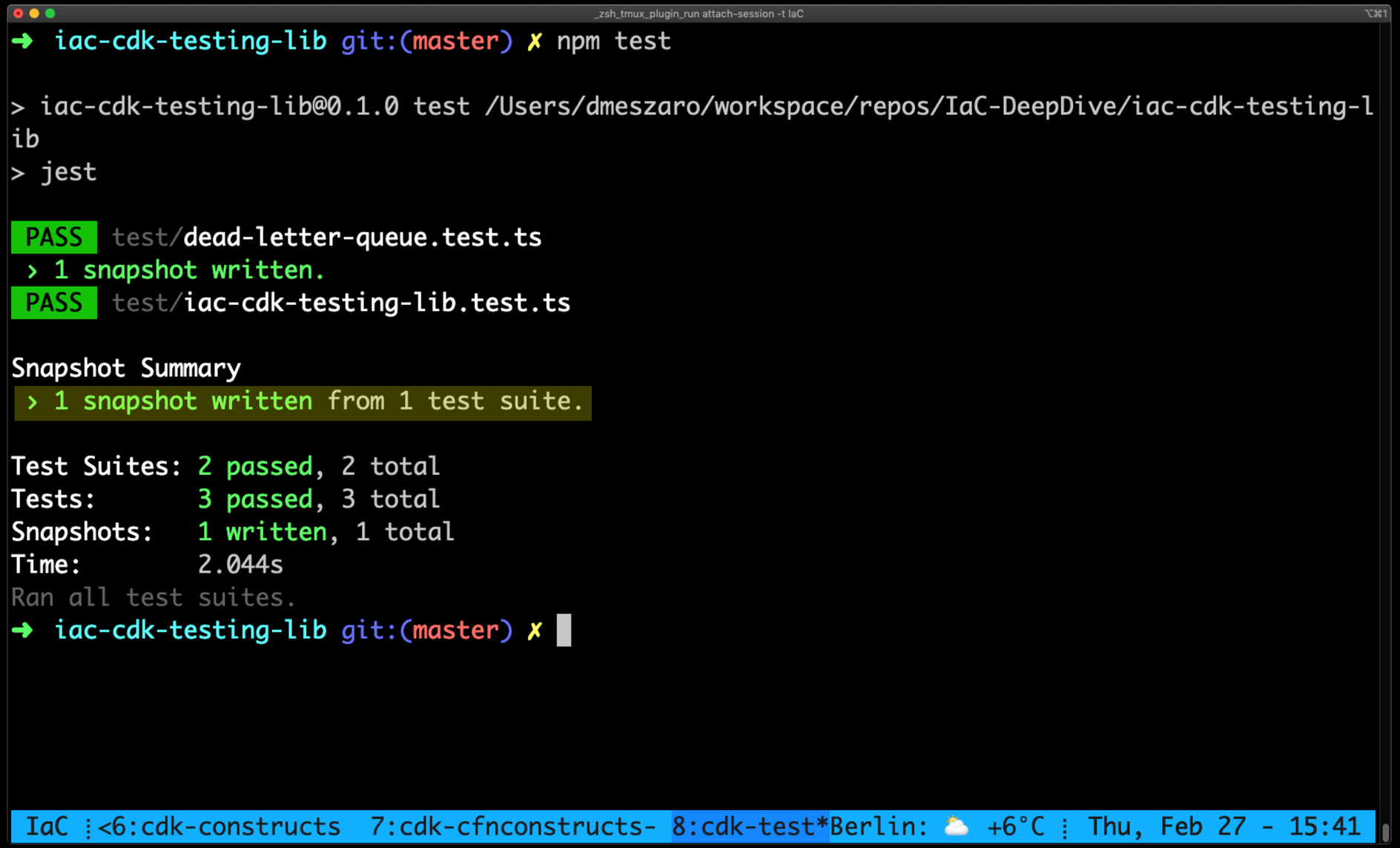
```
npm install --save-dev jest @types/jest @aws-cdk/assert
```

Snapshots

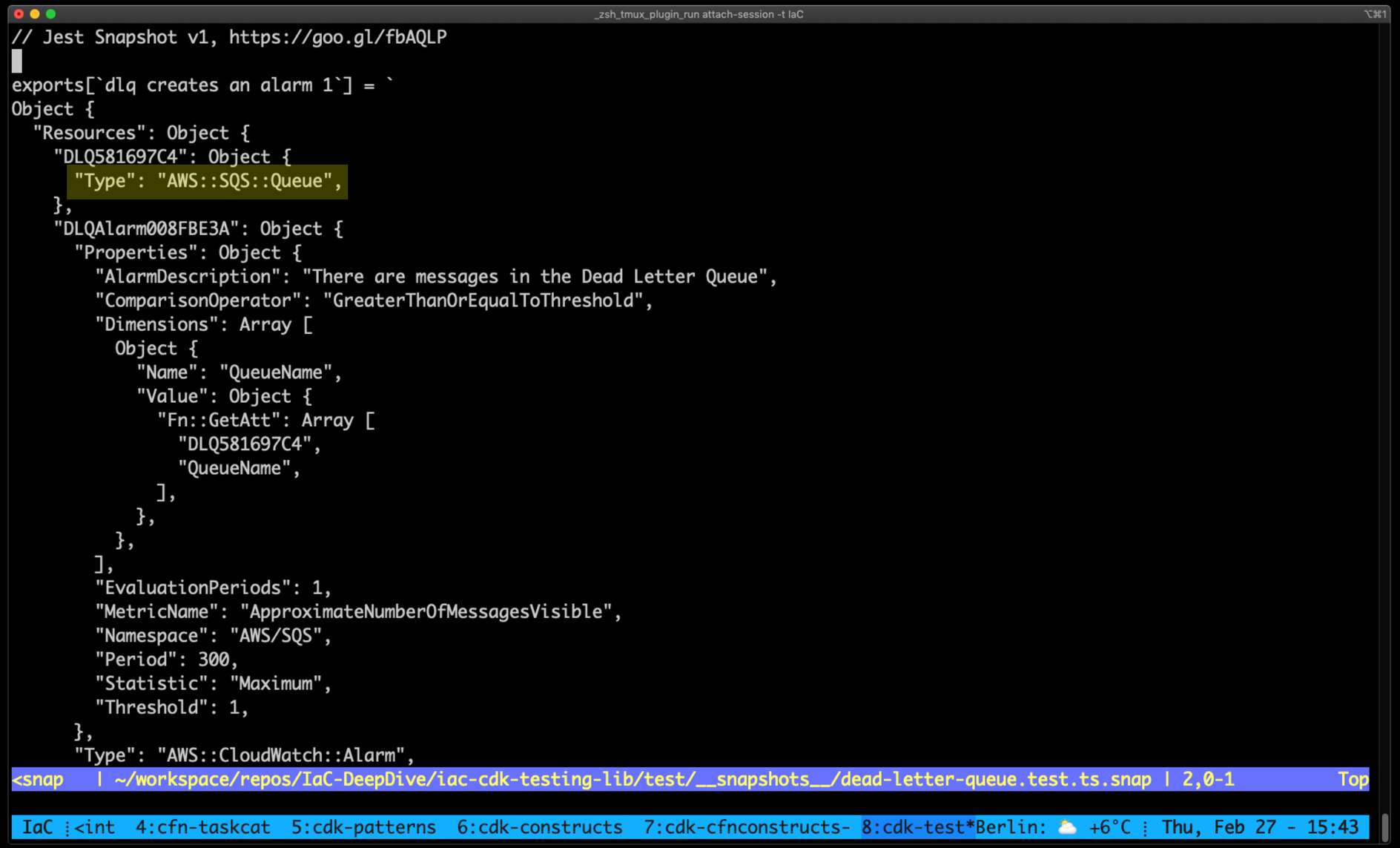
Match my snapshot



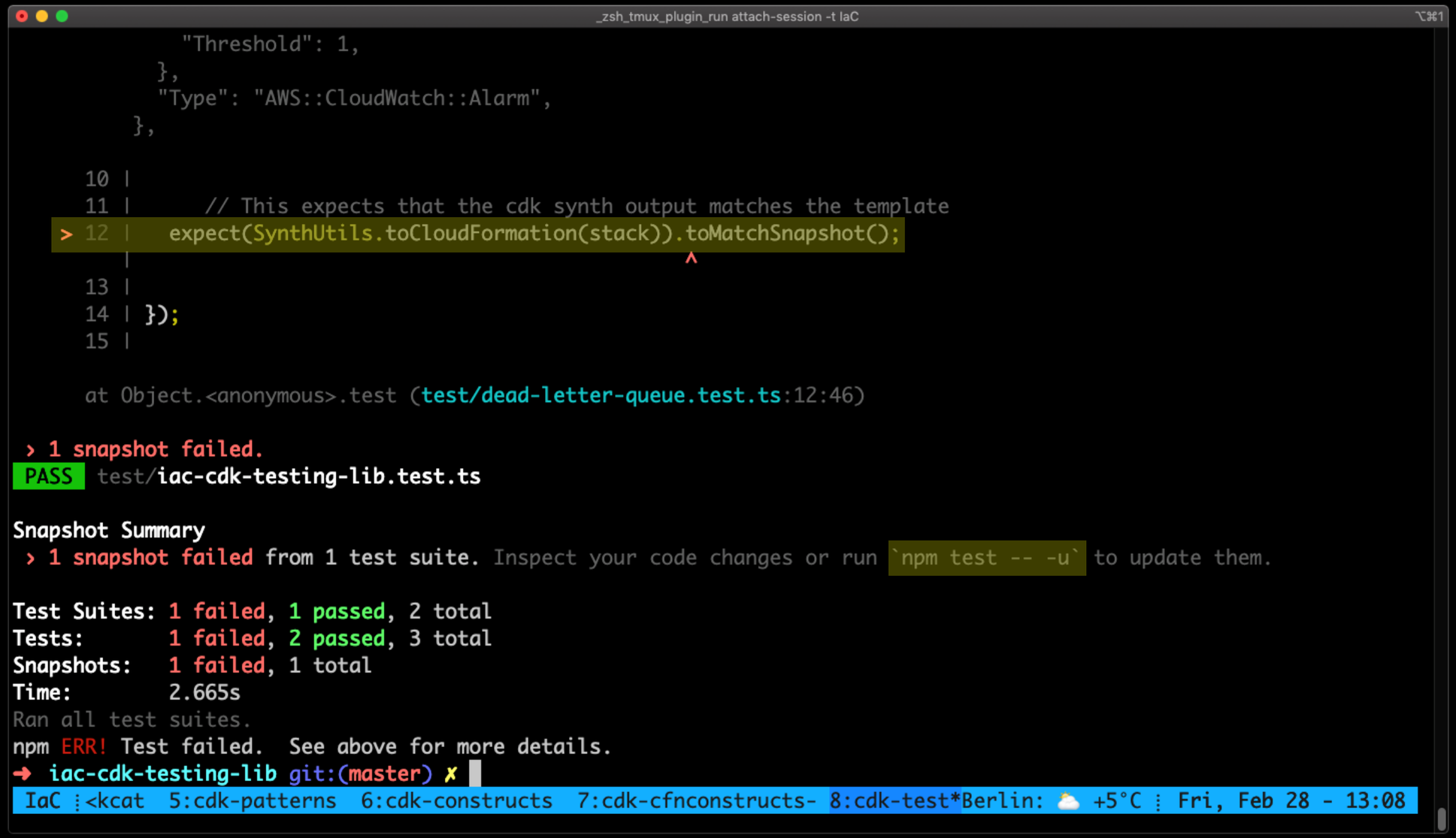
How’s my snapshot?



How’s my snapshot?

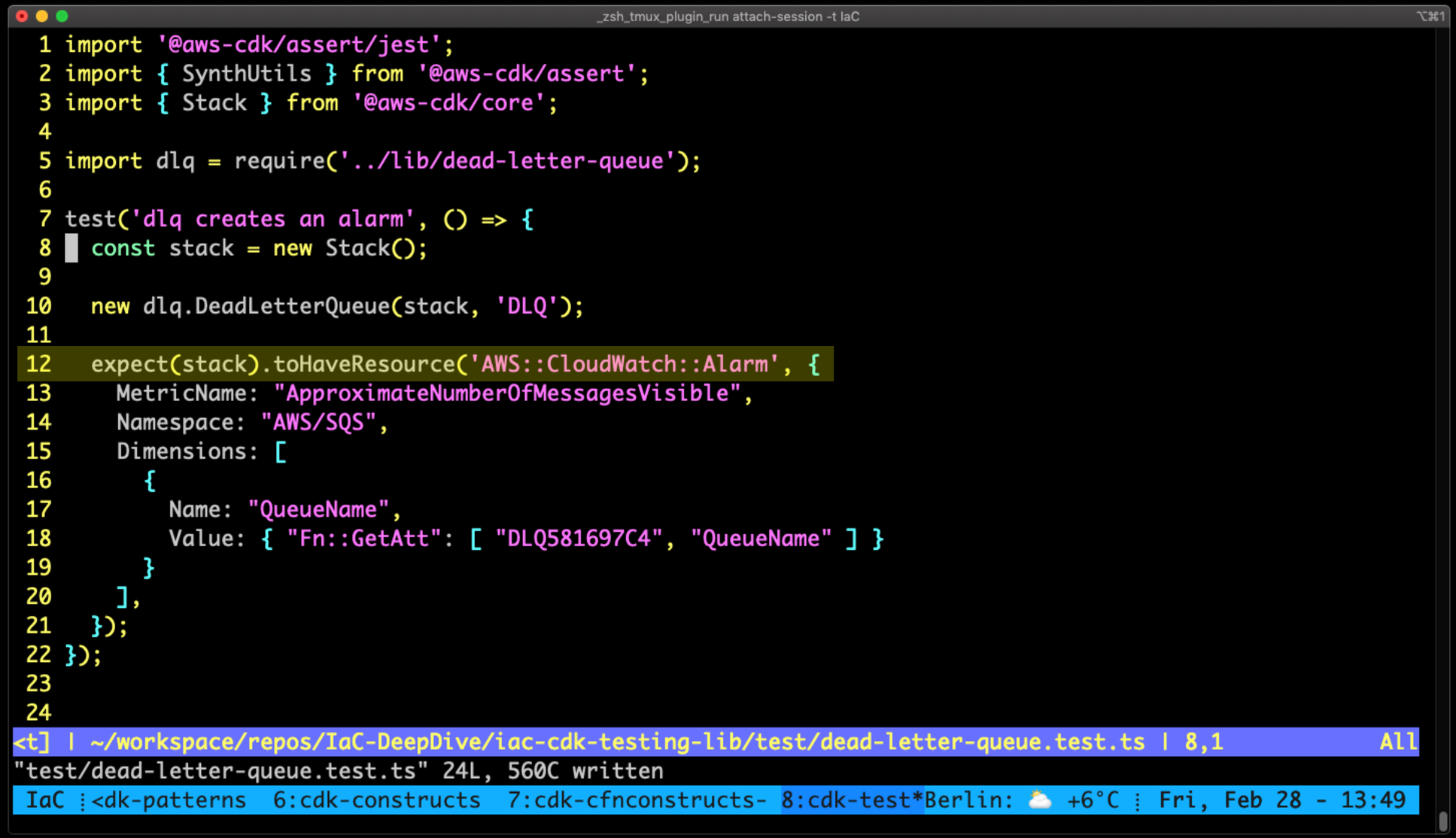


Not matching a snapshot

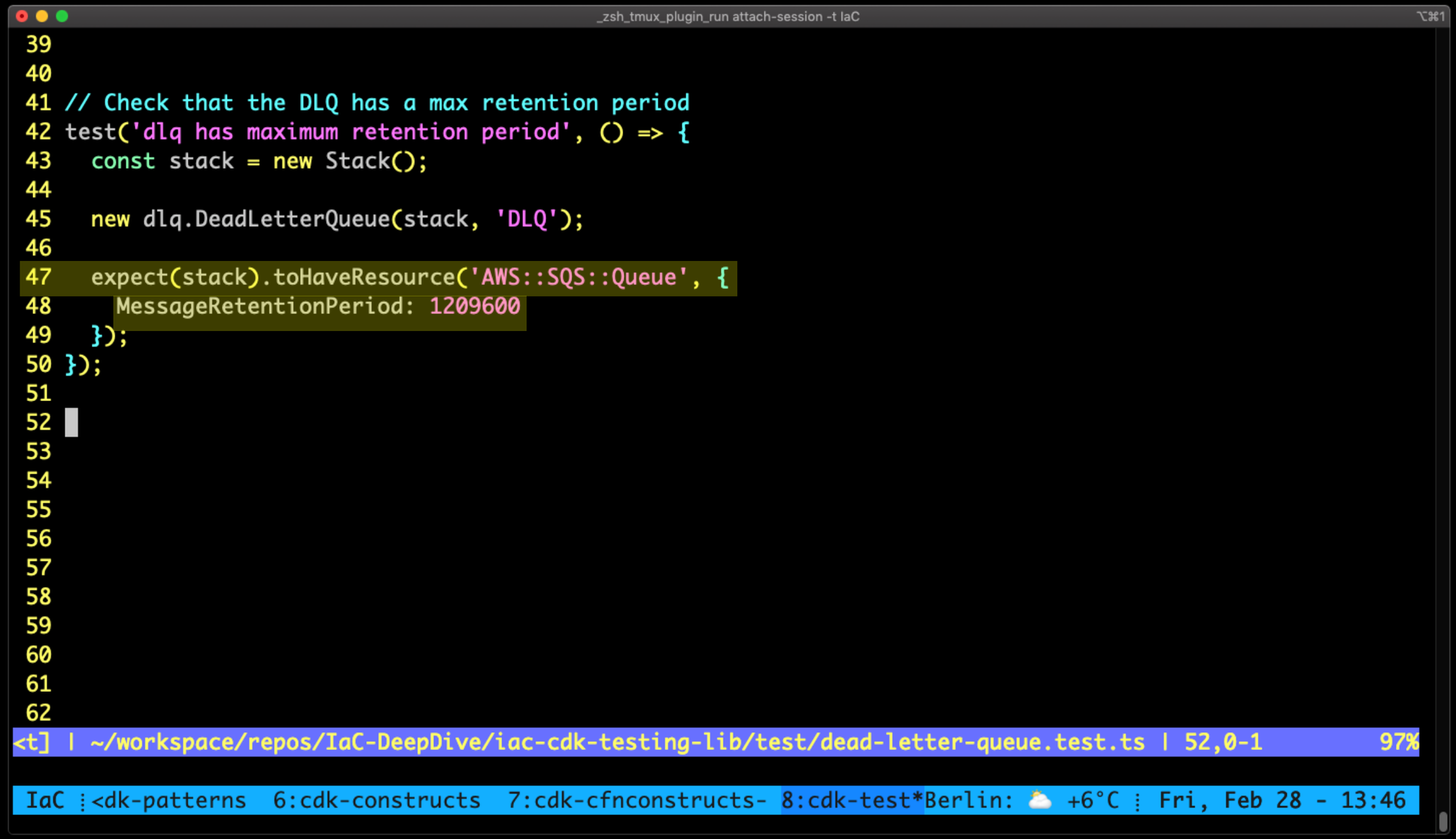


But I want to have more fine-grained control

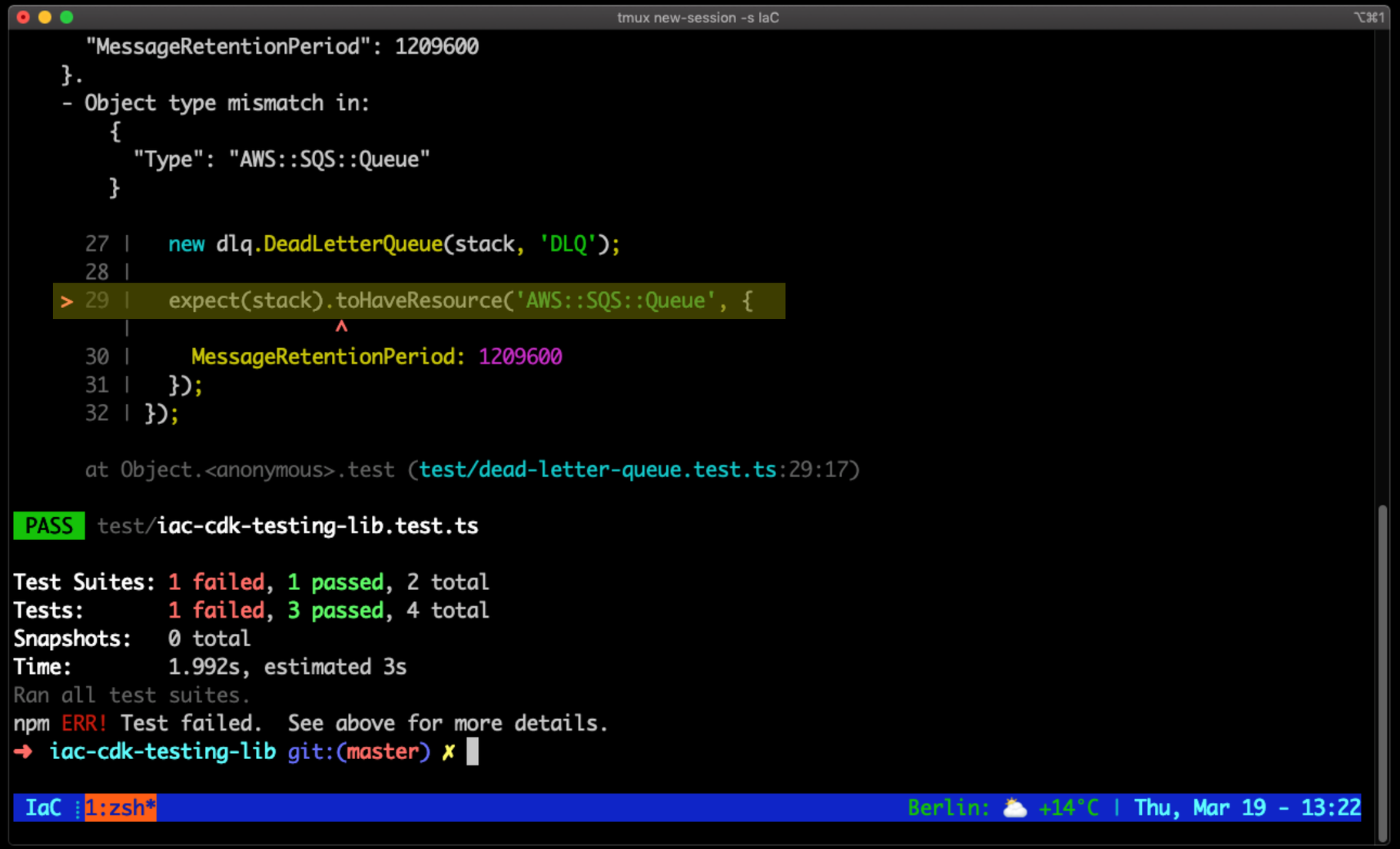
I expect to have this



And this


_zsh_tmux_plugin_run attach-session -t IaC
39
40
41 // Check that the DLQ has a max retention period
42 test('dlq has maximum retention period', () => {
43   const stack = new Stack();
44
45   new dlq.DeadLetterQueue(stack, 'DLQ');
46
47   expect(stack).toHaveResource('AWS::SQS::Queue', {
48   MessageRetentionPeriod: 1209600
49   });
50 });
51
52
53
54
55
56
57
58
59
60
61
62
<t] | ~/workspace/repos/IaC-DeepDive/iac-cdk-testing-lib/test/dead-letter-queue.test.ts | 52,0-1 97%
IaC ⋮<dk-patterns 6:cdk-constructs 7:cdk-cfnconstructs- 8:cdk-test*Berlin: +6°C ⋮ Fri, Feb 28 - 13:46

Then I fail my specific test



How do we do testing with AWS CDK?

- Snapshot tests
- Fine-grained assertions
- Validation tests



```
npm install --save-dev jest @types/jest @aws-cdk/assert
```

You did mention pipelines

Treating infrastructure code the same as any other code is important!

Moving our code down the pipeline

```python
class CdkPythonAppStack(core.Stack):

    def __init__(self, scope: core.Construct, id: str, **kwargs) -> None:
        super().__init__(scope, id, **kwargs)

        # The code that defines your stack goes here
        myVpc = ec2.Vpc(
            self, 'theVpc',
            cidr='10.0.0.0/16',
            )

        # We need more VPCs
        secondVpc = ec2.Vpc(self, 'secondVpc')

        core.CfnOutput(self, 'vpc_id', value=myVpc.id)
        core.CfnOutput(self, 'vpc2_id', value=secondVpc.id)
```



Committing our code to Git

Moving our code down the pipeline

```python
class CdkPythonAppStack(core.Stack):

    def __init__(self, scope: core.Construct, id: str, **kwargs) -> None:
        super().__init__(scope, id, **kwargs)

        # The code that defines your stack goes here
        myVpc = ec2.Vpc(
            self, 'theVpc',
            cidr='10.0.0.0/16',
            )

        # We need more VPCs
        secondVpc = ec2.Vpc(self, 'secondVpc')

        core.CfnOutput(self, 'vpc_id', value=myVpc.id)
        core.CfnOutput(self, 'vpc2_id', value=secondVpc.id)
```



Packaging, linting, syntax checking

Moving our code down the pipeline

```
class CdkPythonAppStack(core.Stack):

    def __init__(self, scope: core.Construct, id: str, **kwargs) -> None:
        super().__init__(scope, id, **kwargs)

        # The code that defines your stack goes here
        myVpc = ec2.Vpc(
            self, 'theVpc',
            cidr='10.0.0.0/16',
            )

        # We need more VPCs
        secondVpc = ec2.Vpc(self, 'secondVpc')

        core.CfnOutput(self, 'vpc_id', value=myVpc.id)
        core.CfnOutput(self, 'vpc2_id', value=secondVpc.id)
```





AWS Lambda



Moving our code down the pipeline

```python
class CdkPythonAppStack(core.Stack):

    def __init__(self, scope: core.Construct, id: str, **kwargs) -> None:
        super().__init__(scope, id, **kwargs)

        # The code that defines your stack goes here
        myVpc = ec2.Vpc(
            self, 'theVpc',
            cidr='10.0.0.0/16',
            )

        # We need more VPCs
        secondVpc = ec2.Vpc(self, 'secondVpc')

        core.CfnOutput(self, 'vpc_id', value=myVpc.id)
        core.CfnOutput(self, 'vpc2_id', value=secondVpc.id)
```





AWS Lambda



Moving our code down the pipeline

```python
class CdkPythonAppStack(core.Stack):

    def __init__(self, scope: core.Construct, id: str, **kwargs) -> None:
        super().__init__(scope, id, **kwargs)

        # The code that defines your stack goes here
        myVpc = ec2.Vpc(
            self, 'theVpc',
            cidr='10.0.0.0/16',
            )

        # We need more VPCs
        secondVpc = ec2.Vpc(self, 'secondVpc')

        core.CfnOutput(self, 'vpc_id', value=myVpc.id)
        core.CfnOutput(self, 'vpc2_id', value=secondVpc.id)
```

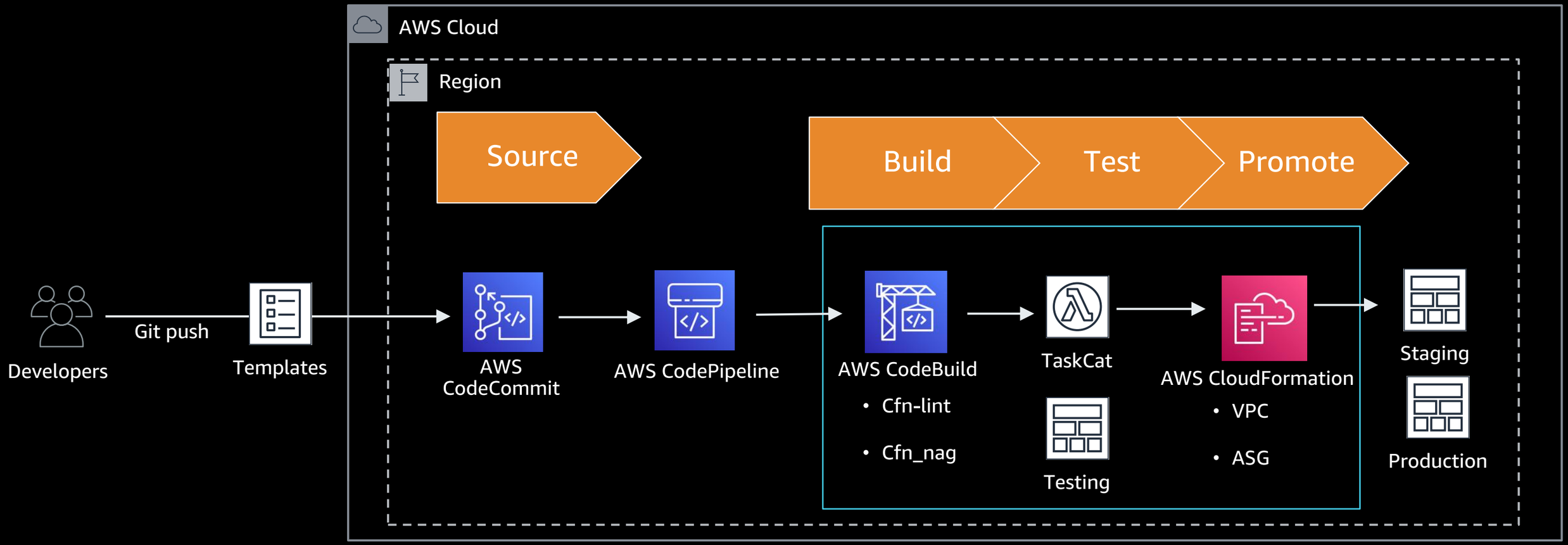


AWS Lambda



Source → Build → Test → Production

AWS CloudFormation: Infrastructure CI/CD



Test me some code!



AWS re:Invent

© 2020, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Homework

- Spice up your toolkit with the right tools to help you write, debug, and test infrastructure code
- Make your next infrastructure deploy be via CodePipeline
- Have fun writing good and secure infrastructure code

And remember . . .



“~~Measure twice, cut once.~~
Test twice, deploy once.”

Unknown

© 2020, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Thank you!

Darko Meszaros

Developer Advocate
AWS

@darkosubotica
ln/darko-mesaros
twitch.tv/ruptwelve
youtube.com/ruptwelve

© 2020, Amazon Web Services, Inc. or its affiliates. All rights reserved.



Please complete the session survey

AWS re:Invent

© 2020, Amazon Web Services, Inc. or its affiliates. All rights reserved.