



AWS Events
93.6K subscribers

Subscribe

32



Share

Download

Clip



1,363 views Jan 9, 2022 [AWS Summit Online ANZ 2021](#)

AWS Cloud Development Kit (CDK) gives you the expressive power of programming languages for defining infrastructure. In this session, we examine how the ancient concept of a test driven development can be applicable to modern software delivery. We guide you through advanced CDK concepts, such as Aspects and level 3 constructs refactoring, as well as some great practices you can apply immediately to your own applications.

DEV05

Driving a test-first strategy with CDK and test driven development

Nelli Lovchikova
Solutions Architect
Amazon Web Services

AWS CDK

A multilanguage development framework for modeling infrastructure as reusable components



```
class UriShortener extends Stack {  
  constructor(scope: App, id: string, props?: UriShortenerProps) {  
    super(scope, id, props);  
  
    const vpc = new ec2.Vpc(this, 'vpc', { maxAzs: 2 });  
    const cluster = new ecs.Cluster(this, 'cluster', { vpc: vpc });  
    const service = new patterns.NetworkLoadBalancedFargateService(this, 'sample-app', {  
      cluster,  
      taskImageOptions: {  
        image: ecs.ContainerImage.fromAsset('ping'),  
      },  
    },  
    dom  
  );  
  
  // Setup AutoScaling policy  
  const scaling = service.service.autoScaleTask;  
  scaling.scaleOnCpuUtilization('CpuScaling', {  
    targetUtilizationPercent: 50,  
    scaleInCooldown: Duration.seconds(60),  
    scaleOutCooldown: Duration.seconds(60)  
  });  
}
```



GO
(Alpha
preview)

AWS CDK

CONCEPTS

Stack – unit of deployment

Construct – component

Resource – AWS Resource

Aspects – the way to apply operation to all constructs in a scope

“Software never was perfect and won't get perfect. But is that a license to create garbage?”

Boris Beizer

Software Engineer and author of
Software Testing Techniques

Test Driven Development (TDD)

Unit tests is a code that sets up inputs and validates outputs of a function, **TDD** is about designing business logic using unit tests as a driver.

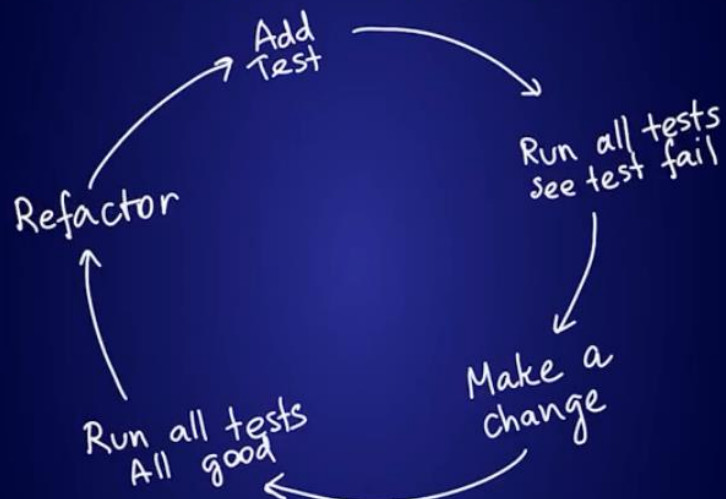
Test Driven Development (TDD)

BENEFITS

- Reduced bug density
- Design time observability
- Easy refactoring
- Details code documentation

Test Driven Development (TDD)

FLOW



The story



Demo explained

Problem

Build a communication channel component for others to use.

Solution

Split problem into small sub-problems

Workflow

Write test for the next sub-problem

Observe it fail

Make it work

Refactor (as needed)

Repeat till the problem is solved

```
cdk init --language typescript -app MyKlingon01
```

Repository <https://bit.ly/38ilkd5>

Demo

TDD workflow with CDK

package.json package.json/...

> bin

> lib

> node_modules

> test

TS tdd-myklingon.test.ts

.gitignore

.npmignore

cdk.json

jest.config.js

package-lock.json

package.json

README.md

tsconfig.json

```
{
  "name": "tdd-myklingon",
  "version": "0.1.0",
  "bin": {
    "tdd-myklingon": "bin/tdd-myklingon.js"
  },
  "scripts": {
    "build": "tsc",
    "watch": "tsc -w",
    "test": "jest",
    "cdk": "cdk"
  },
  "devDependencies": {
    "@aws-cdk/assert": "1.89.0",
    "@types/jest": "^26.0.10",
    "@types/node": "10.17.27",
    "jest": "^26.4.2",
    "ts-jest": "^26.2.0",
    "aws-cdk": "1.89.0",
    "ts-node": "^9.0.0",
    "typescript": "~3.9.7"
  },
  "dependencies": {
    "@aws-cdk/aws-apigateway": "^1.89.0",
    "@aws-cdk/core": "1.89.0",
    "source-map-support": "^0.5.16"
  }
}
```

TERMINAL

aws summit>npm test

aws summit>git checkout api0
Switched to branch 'api0'
aws summit>

package.json package.json/() devDependencies

```
{
  "name": "tdd-myklingon",
  "version": "0.1.0",
  "bin": {
    "tdd-myklingon": "bin/tdd-myklingon.js"
  },
  "scripts": {
    "build": "tsc",
    "watch": "tsc -w",
    "test": "jest --watchAll",
    "cdk": "cdk"
  },
  "devDependencies": {
    "@aws-cdk/assert": "1.89.0",
    "@types/jest": "^26.0.10",
    "@types/node": "10.17.27",
    "jest": "^26.4.2",
    "ts-jest": "^26.2.0",
    "aws-cdk": "1.89.0",
    "ts-node": "^9.0.0",
    "typescript": "~3.9.7"
  },
  "dependencies": {
    "@aws-cdk/aws-apigateway": "^1.89.0",
    "@aws-cdk/core": "1.89.0",
    "source-map-support": "^0.5.16"
  }
}
```

TERMINAL

aws summit>npm test

aws summit>git checkout api0
Switched to branch 'api0'
aws summit>git checkout api1
Switched to branch 'api1'
aws summit>


```
TS channel.test.ts test/channel.test.ts/ describe('Given that communication channel is needed') callback
import * as cdk from '@aws-cdk/core';
import '@aws-cdk/assert/jest';

describe('Given that communication channel is needed', () => {
  describe('When changes are allowed', () => {
    it('Then requester can create a new record', () => {
      const stack = new cdk.Stack();

      new Channel(stack, 'api', {
        isReadOnly: false
      });

      expect(stack).toHaveResource("AWS::ApiGateway::Method", {
        HttpMethod: 'POST'
      })
    });
  });
});
});
```

aws summit>npm test

aws summit>git checkout api0
Switched to branch 'api0'
aws summit>git checkout api1
Switched to branch 'api1'
aws summit>

```
{ } package.json package.json/...
{
  "name": "tdd-myklingon",
  "version": "0.1.0",
  "bin": {
    "tdd-myklingon": "bin/tdd-myklingon.js"
  },
  "scripts": {
    "build": "tsc",
    "watch": "tsc -w",
    "test": "jest --watchAll",
    "cdk": "cdk"
  },
  "devDependencies": {
    "@aws-cdk/assert": "1.89.0",
    "@types/jest": "^26.0.10",
    "@types/node": "10.17.27",
    "jest": "^26.4.2",
    "ts-jest": "^26.2.0",
    "aws-cdk": "1.89.0",
    "ts-node": "^9.0.0",
    "typescript": "^3.9.7"
  },
  "dependencies": {
    "@aws-cdk/aws-apigateway": "^1.89.0",
    "@aws-cdk/core": "1.89.0",
    "source-map-support": "^0.5.16"
  }
}
```

PASS test/tdd-myklingon.test.ts
FAIL test/channel.test.ts
● Test suite failed to run

test/channel.test.ts:9:17 - error T52304: Cannot find name 'Channel'.

```
    new Channel(stack, 'a
pi', {
```

Test Suites: 1 failed, 1 passed, 2 total
Tests: 1 passed, 1 total
Snapshots: 0 total
Time: 4.588 s
Ran all test suites.

Watch Usage
 > Press f to run only failed tests.
 > Press o to only run tests related to changed files.
 > Press p to filter by a filename regex pattern.
 > Press t to filter by a test name regex pattern.
 > Press q to quit watch mode.
 > Press Enter to trigger a test run.

aws summit>git checkout api0
Switched to branch 'api0'
aws summit>git checkout api1
Switched to branch 'api1'
aws summit>

```
TS channel.test.ts test/channel.test.ts/*ChannelProps
import * as cdk from '@aws-cdk/core';
import '@aws-cdk/assert/jest';

describe('Given that communication channel is needed', () => {
  describe('When changes are allowed', () => {
    it('Then requester can create a new record', () => {
      const stack = new cdk.Stack();

      new Channel(stack, 'api', {
        readOnly: false
      });

      expect(stack).toHaveResource("AWS::ApiGateway::Method", {
        HttpMethod: 'POST'
      });
    });
  });
});
```

TERMINAL ... 1: pwsh, pwsh +

```
PASS test/tdd-myklingon.test.ts
FAIL test/channel.test.ts
  ● Test suite failed to run

    test/channel.test.ts:9:17 - error TS2304: Cannot find name 'Channel'.

    9       new Channel(stack, 'a
      pi', {
        // ...
      });
    ~~~~~

Test Suites: 1 failed, 1 passed, 2 total
Tests: 1 passed, 1 total
Snapshots: 0 total
Time: 4.588 s
Ran all test suites.

Watch Usage
  > Press f to run only failed tests.
  > Press o to only run tests related to
    changed files.
  > Press p to filter by a filename reg
    ex pattern.
  > Press t to filter by a test name reg
    ex pattern.
  > Press q to quit watch mode.
  > Press Enter to trigger a test run.

aws summit>git checkout api0
Switched to branch 'api0'
aws summit>git checkout api1
Switched to branch 'api1'
aws summit>
```

We now need to make this green

```
TS channel.test.ts on channel is needed' callback/ describe('When changes are allowed' callback/ it('Then requester can create a new record' callback
import * as cdk from '@aws-cdk/core';
import '@aws-cdk/assert/jest';
import * as apigateway from '@aws-cdk/aws-apigateway';

interface ChannelProps{
  readOnly: boolean
}

class Channel extends cdk.Construct{
  constructor(scope: cdk.Construct, id: string, props: ChannelProps) {
    super(scope, id);
    const api = new apigateway.RestApi(this, 'books-api');
    api.root.addMethod('POST');
  }
}

describe('Given that communication channel is needed', () => {
  describe('When changes are allowed', () => {
    it('Then requester can create a new record', () => {
      const stack = new cdk.Stack();

      new Channel(stack, 'api', {
        readOnly: false
      });

      expect(stack).toHaveResource("AWS::ApiGateway::Method", {
        HttpMethod: 'POST'
      });
    });
  });
});
```

TERMINAL ... 1: pwsh, pwsh +

```
PASS test/tdd-myklingon.test.ts
PASS test/channel.test.ts

Test Suites: 2 passed, 2 total
Tests: 2 passed, 2 total
Snapshots: 0 total
Time: 7.881 s
Ran all test suites.

Watch Usage: Press w to show more.

aws summit>git checkout api0
Switched to branch 'api0'
aws summit>git checkout api1
Switched to branch 'api1'
aws summit>git checkout api2
Switched to branch 'api2'
aws summit>
```

Let's add more tests

```
TS channel.test.ts test/channel.test.ts describe('Given that communication channel is needed') callback

import * as cdk from '@aws-cdk/core';
import '@aws-cdk/assert/jest';
import * as apigateway from '@aws-cdk/aws-apigateway';

interface ChannelProps {
  isReadOnly: boolean
}

class Channel extends cdk.Construct {
  constructor(scope: cdk.Construct, id: string, props: ChannelProps) {
    super(scope, id);
    const api = new apigateway.RestApi(this, 'books-api');
    api.root.addMethod('POST');
  }
}

describe('Given that communication channel is needed', () => {
  let stack: cdk.Stack;

  beforeEach(() => {
    stack = new cdk.Stack();
  });

  describe('When changes are allowed', () => {
    it('Then requester can create a new record', () => {
      const stack = new cdk.Stack();

      new Channel(stack, 'api', {
        isReadOnly: false
      });

      expect(stack).toHaveResource("AWS::ApiGateway::Method", {
        HttpMethod: 'POST'
      });
    });
  });
});
```

Brush Test (B)

TERMINAL 1: pwsh, pwsh

RUNS test/channel.test.ts
RUNS test/tdd-myklingon.test.ts

Test Suites: 0 of 2 total
Tests: 0 total
Snapshots: 0 total
Time: 3 s, estimated 5 s

aws summit>git checkout api0
Switched to branch 'api0'
aws summit>git checkout api1
Switched to branch 'api1'
aws summit>git checkout api2
Switched to branch 'api2'
aws summit>git checkout api3
Switched to branch 'api3'
aws summit>

```
TS channel.test.ts test/channel.test.ts describe('Given that communication channel is needed') callback

}

class Channel extends cdk.Construct {
  constructor(scope: cdk.Construct, id: string, props: ChannelProps) {
    super(scope, id);
    const api = new apigateway.RestApi(this, 'books-api');
    api.root.addMethod('POST');
  }
}

describe('Given that communication channel is needed', () => {
  let stack: cdk.Stack;

  beforeEach(() => {
    stack = new cdk.Stack();
  });

  describe('When changes are allowed', () => {
    it('Then requester can create a new record', () => {
      const stack = new cdk.Stack();

      new Channel(stack, 'api', {
        isReadOnly: false
      });

      expect(stack).toHaveResource("AWS::ApiGateway::Method", {
        HttpMethod: 'POST'
      });
    });
  });

  describe('When changes are not allowed', () => {
    it('Then requester cannot create a new record', () => {
      new Channel(stack, 'api', {
        isReadOnly: true
      });
    });
  });
});
```

TERMINAL 1: pwsh, pwsh

PASS test/tdd-myklingon.test.ts
FAIL test/channel.test.ts

● Given that communication channel is needed › When changes are not allowed › Then requester cannot create a new record

Not None of 0 resources matches resource 'AWS::ApiGateway::Method' with {
 "\$objectLike": {
 "HttpMethod": "POST"
 }
}.

```
41 |         isReadOnly:
true 42 |         });
> 43 |         expect(stack).not.toHaveResource("AWS::ApiGateway::Method", {
    |                               ^
44 |           HttpMethod:
'POST' 45 |         });
      46 |       });
      at Object.<anonymous> (test/channel.test.ts:43:31)
```

Test Suites: 1 failed, 1 passed, 2 total
1

aws summit>git checkout api0
Switched to branch 'api0'
aws summit>git checkout api1
Switched to branch 'api1'
aws summit>git checkout api2
Switched to branch 'api2'
aws summit>git checkout api3
Switched to branch 'api3'
aws summit>

```
TS channel.test.ts test/channel.test.ts describe('Given that communication channel is needed') callback

const stack = new cdk.Stack();

new Channel(stack, 'api', {
  isReadOnly: false
});

expect(stack).toHaveResource("AWS::ApiGateway::Method", {
  HttpMethod: 'POST'
});
});

describe('When changes are not allowed', () => {
  it('Then requester cannot create a new record', () => {
    new Channel(stack, 'api', {
      isReadOnly: true
    });
    expect(stack).not.toHaveResource("AWS::ApiGateway::Method", {
      HttpMethod: 'POST'
    });
  });
});
});

Not None of 0 resources matches resource 'AWS::ApiGateway::Method' with {
  "ObjectLike": {
    "HttpMethod": "POST"
  }
}.
41 | isReadOnly:
42 | });
43 | expect(stack).not.toHaveResource("AWS::ApiGateway::Method", {
44 |   HttpMethod:
45 |   });
46 | });
at Object.<anonymous> (test/channel.test.ts:43:31)

Test Suites: 1 failed, 1 passed, 2 total
Tests: 1 failed, 2 passed, 3 total
Snapshots: 0 total
Time: 6.263 s
Ran all test suites.

Watch Usage: Press w to show more.

aws summit>git checkout api0
Switched to branch 'api0'
aws summit>git checkout api1
Switched to branch 'api1'
aws summit>git checkout api2
Switched to branch 'api2'
aws summit>git checkout api3
Switched to branch 'api3'
aws summit>
```

Now let's fix the error

```
TS channel.test.ts test/channel.test.ts describe('Given that communication channel is needed') callback

}

class Channel extends cdk.Construct {
  constructor(scope: cdk.Construct, id: string, props: ChannelProps) {
    super(scope, id);
    const api = new apigateway.RestApi(this, 'books-api');
    api.root.addMethod('GET');
    if (!props.isReadOnly) {
      api.root.addMethod('POST');
    }
  }
}

describe('Given that communication channel is needed', () => {
  let stack: cdk.Stack;

  beforeEach(() => {
    stack = new cdk.Stack();
  });

  describe('When changes are allowed', () => {
    it('Then requester can create a new record', () => {
      const stack = new cdk.Stack();

      new Channel(stack, 'api', {
        isReadOnly: false
      });

      expect(stack).toHaveResource("AWS::ApiGateway::Method", {
        HttpMethod: 'POST'
      });
    });
  });

  describe('When changes are not allowed', () => {
    it('Then requester cannot create a new record', () => {
      const stack = new cdk.Stack();

      new Channel(stack, 'api', {
        isReadOnly: true
      });

      expect(stack).not.toHaveResource("AWS::ApiGateway::Method", {
        HttpMethod: 'POST'
      });
    });
  });
});

PASS test/tdd-myklingon.test.ts
PASS test/channel.test.ts

Test Suites: 2 passed, 2 total
Tests: 3 passed, 3 total
Snapshots: 0 total
Time: 7.049 s
Ran all test suites.

Watch Usage: Press w to show more.

aws summit>git checkout api0
Switched to branch 'api0'
aws summit>git checkout api1
Switched to branch 'api1'
aws summit>git checkout api2
Switched to branch 'api2'
aws summit>git checkout api3
Switched to branch 'api3'
aws summit>git checkout api4
Switched to branch 'api4'
aws summit>
```



```
TS channel.test.ts test/channel.test.ts Channel/ constructor
}

class Channel extends cdk.Construct {
  constructor(scope: cdk.Construct, id: string, props: ChannelProps) {
    super(scope, id);
    const api = new apigateway.RestApi(this, 'books-api');
    //api.root.addMethod('GET');
    if (!props.isReadOnly) {
      api.root.addMethod('POST');
    }
  }
}

describe('Given that communication channel is needed', () => {
  let stack: cdk.Stack;

  beforeEach(() => {
    stack = new cdk.Stack();
  });

  describe('When changes are allowed', () => {
    it('Then requester can create a new record', () => {
      const stack = new cdk.Stack();

      new Channel(stack, 'api', {
        isReadOnly: false
      });

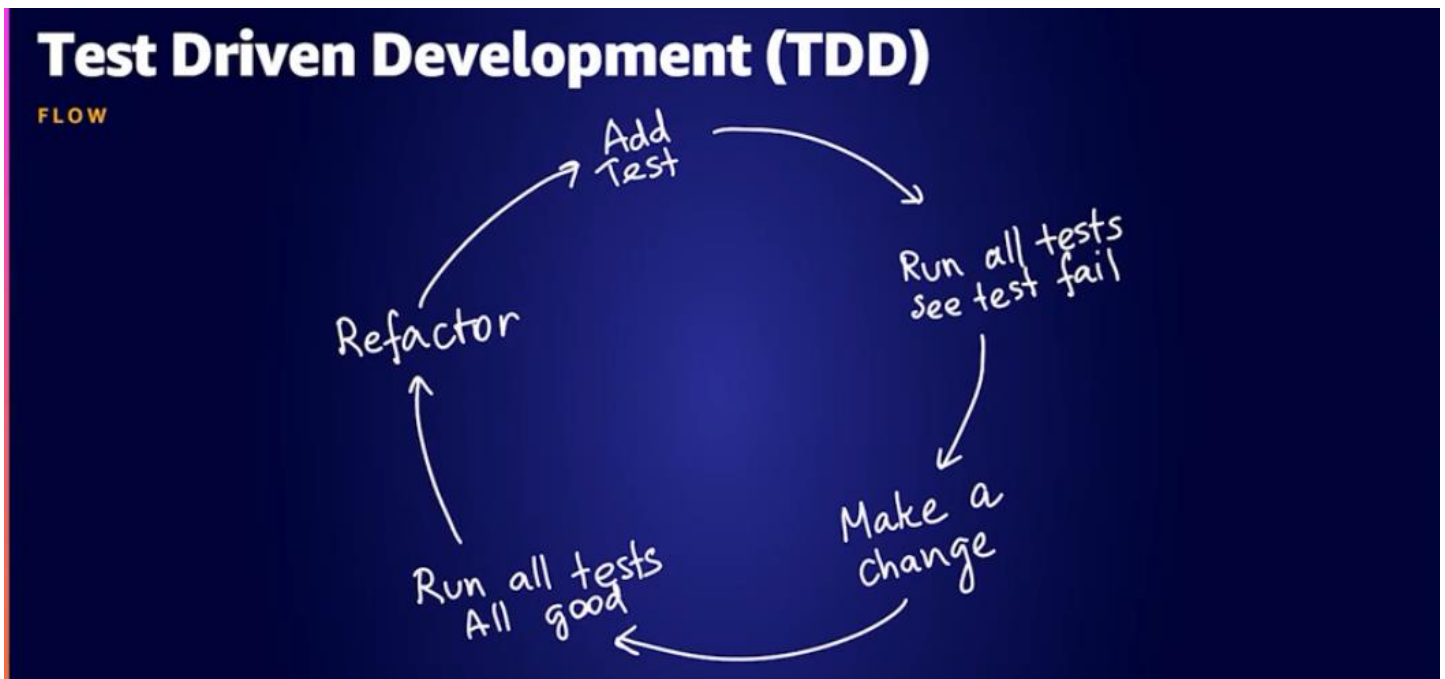
      expect(stack).toHaveResource("AWS::ApiGateway::Method", {
        HttpMethod: 'POST'
      });
    });
  });

  describe('When changes are not allowed', () => {
    it('Then requester cannot create a new record', () => {

```

```
TERMINAL 1: pwsh, pwsh
PASS test/tdd-myklington.test.ts
FAIL test/channel.test.ts
  Given that communication channel is needed: When changes are not allowed
    Then requester cannot create a new record
      Validation failed with the following errors:
        [Default/api/books-api] The REST API doesn't contain any methods
      true
      44 | isReadOnly:
      45 | });
      46 | expect(stack).no
t.toHaveResource("AWS::ApiGateway::Method", {
      47 | HttpMethod:
      48 | 'POST'
      49 | });
      at validateTree (node_modules/@aws-cdk/core/lib/private/synthesis.ts:182:11)
      at Object.synthesize (node_modules/@aws-cdk/core/lib/private/synthesis.ts:28:5)
      at App.synth (node_modules/@aws-cdk
aws summit>git checkout api0
Switched to branch 'api0'
aws summit>git checkout api1
Switched to branch 'api1'
aws summit>git checkout api2
Switched to branch 'api2'
aws summit>git checkout api3
Switched to branch 'api3'
aws summit>git checkout api4
Switched to branch 'api4'
aws summit>
```

The test fails if we comment out the addMethod as above



Demo 2

```
{ } package.json package.json/...
{
  "name": "tdd-myklingon",
  "version": "0.1.0",
  "bin": {
    "tdd-myklingon": "bin/tdd-myklingon.js"
  },
  "scripts": {
    "build": "tsc",
    "watch": "tsc -w",
    "test": "jest",
    "cdk": "cdk"
  },
  "devDependencies": {
    "@aws-cdk/assert": "1.89.0",
    "@types/jest": "^26.0.10",
    "@types/node": "10.17.27",
    "jest": "^26.4.2",
    "ts-jest": "^26.2.0",
    "aws-cdk": "1.89.0",
    "ts-node": "^9.0.0",
    "typescript": "~3.9.7"
  },
  "dependencies": {
    "@aws-cdk/aws-iam": "^1.90.0",
    "@aws-cdk/core": "1.89.0",
    "source-map-support": "^0.5.16"
  }
}
```

```
test/permissionCheck.test.ts:19:43 - error TS2304:
Cannot find name 'PermissionCheck'.
19      cdk.Aspects.of(stack).add(new Permi
ssionCheck());
20
test/permissionCheck.test.ts:21:13 - error TS1109:
Expression expected.
21      ???
22
test/permissionCheck.test.ts:21:15 - error TS1109:
Expression expected.
21      ???
22
test/permissionCheck.test.ts:22:9 - error TS1109:
Expression expected.
22      });
23
Test Suites: 1 failed, 1 passed, 2 total
Tests: 1 passed, 1 total
Snapshots: 0 total
Time: 4.057 s
Ran all test suites.

Watch Usage: Press w to show more.

aws summit>git checkout asp0
Switched to branch 'asp0'
aws summit>
```

We now have an IAM dependency added and we are using Aspects as below

```
TS permissionCheck.test.ts test/permissionCheck.test.ts/ describe('Given that scope is provided') callback/ describe('When there is an allow policy')
import * as cdk from '@aws-cdk/core';
import '@aws-cdk/assert/jest';
import * as iam from '@aws-cdk/aws-iam';

describe('Given that scope is provided', () => {
  describe('When there is an allow policy', () => {
    test('and no resource restrictions then warning is added', () => {
      const stack = new cdk.Stack();

      const role = new iam.Role(stack, 'myrole.iamrole', {
        assumedBy: new iam.ServicePrincipal('sns.amazonaws.com'),
      });

      role.addToPolicy(new iam.PolicyStatement({
        resources: ['*'],
        actions: ['lambda:InvokeFunction'],
      }));

      cdk.Aspects.of(stack).add(new PermissionCheck());

      ???
    });
  });
});
```

```
test/permissionCheck.test.ts:19:43 - error TS2304:
Cannot find name 'PermissionCheck'.
19      cdk.Aspects.of(stack).add(new Permi
ssionCheck());
20
test/permissionCheck.test.ts:21:13 - error TS1109:
Expression expected.
21      ???
22
test/permissionCheck.test.ts:21:15 - error TS1109:
Expression expected.
21      ???
22
test/permissionCheck.test.ts:22:9 - error TS1109:
Expression expected.
22      });
23
Test Suites: 1 failed, 1 passed, 2 total
Tests: 1 passed, 1 total
Snapshots: 0 total
Time: 4.057 s
Ran all test suites.

Watch Usage: Press w to show more.

aws summit>git checkout asp0
Switched to branch 'asp0'
aws summit>
```

```
TS permissionCheck.test.ts test/permissionCheck.test.ts/...
import * as cdk from '@aws-cdk/core';
import '@aws-cdk/assert/jest';
import * as iam from '@aws-cdk/aws-iam';
import { SynthUtils } from '@aws-cdk/assert';

describe('Given that scope is provided', () => {
  describe('When there is an allow policy', () => {
    test('and no resource restrictions then warning is added', () => {
      const stack = new cdk.Stack();

      const role = new iam.Role(stack, 'myrole.iamrole', {
        assumedBy: new iam.ServicePrincipal('sns.amazonaws.com'),
      });

      role.addToPolicy(new iam.PolicyStatement({
        resources: ['*'],
        actions: ['lambda:InvokeFunction'],
      }));

      cdk.Aspects.of(stack).add(new PermissionCheck());

      expect(stack).toHaveWarning('Wildcard resources aren\'t allowed');
    });
  });
});

declare global {
  namespace jest {
    interface Matchers<R> {
      toHaveWarning(message: string): R;
    }
  }
}

expect.extend({
  toHaveWarning(received: cdk.Stack, message) {
    const synthResult = SynthUtils.synthesize(received);
```

```
TERMINAL 1: pwsh, pwsh
RUNS test/tdd-myklingon.test.ts
RUNS test/permissionCheck.test.ts

Test Suites: 0 of 2 total
Tests: 0 total
Snapshots: 0 total
Time: 1 s, estimated 2 s

aws summit>git checkout asp0
Switched to branch 'asp0'
aws summit>git checkout asp1
Switched to branch 'asp1'
aws summit>
```

```
TS permissionCheck.test.ts test/permissionCheck.test.ts/...
import * as cdk from '@aws-cdk/core';
import '@aws-cdk/assert/jest';
import * as iam from '@aws-cdk/aws-iam';
import { SynthUtils } from '@aws-cdk/assert';

describe('Given that scope is provided', () => {
  describe('When there is an allow policy', () => {
    test('and no resource restrictions then warning is added', () => {
      const stack = new cdk.Stack();

      const role = new iam.Role(stack, 'myrole.iamrole', {
        assumedBy: new iam.ServicePrincipal('sns.amazonaws.com'),
      });

      role.addToPolicy(new iam.PolicyStatement({
        resources: ['*'],
        actions: ['lambda:InvokeFunction'],
      }));

      cdk.Aspects.of(stack).add(new PermissionCheck());

      expect(stack).toHaveWarning('Wildcard resources aren\'t allowed');
    });
  });
});

declare global {
  namespace jest {
    interface Matchers<R> {
      toHaveWarning(message: string): R;
    }
  }
}

expect.extend({
  toHaveWarning(received: cdk.Stack, message) {
    const synthResult = SynthUtils.synthesize(received);
```

```
TERMINAL 1: pwsh, pwsh
PASS test/tdd-myklingon.test.ts
FAIL test/permissionCheck.test.ts
  ● Test suite failed to run

    test/permissionCheck.test.ts:20:43 - error TS2304:
    Cannot find name 'PermissionCheck'.

    20 cdk.Aspects.of(stack).add(new Permi
    ssionCheck());
    ~~~~~

Test Suites: 1 failed, 1 passed, 2 total
Tests: 1 passed, 1 total
Snapshots: 0 total
Time: 4.761 s
Ran all test suites.

Watch Usage: Press w to show more.
```

```
aws summit>git checkout asp0
Switched to branch 'asp0'
aws summit>git checkout asp1
Switched to branch 'asp1'
aws summit>
```



```
TS permissionCheck.test.ts test/permissionCheck.test.ts/...
    });
    cdk.Aspects.of(stack).add(new PermissionCheck());

    expect(stack).toHaveWarning('Wildcard resources aren\'t allowed');
  });
});

declare global {
  namespace jest {
    interface Matchers<R> {
      toHaveWarning(message: string): R;
    }
  }
}

expect.extend({
  toHaveWarning(received: cdk.Stack, message) {
    const synthResult = SynthUtils.synthesize(received);

    var pass = synthResult.messages
      .filter(x => x.level == 'warning')
      .some(x => x.entry.data == message);
    const output = pass ? 'Template has expected error' : 'Template doesn\'t have an expected error';

    return {
      pass, message: () => output
    }
  }
});

TERMINAL 1: pwsh, pwsh
PASS test/tdd-myklingon.test.ts
FAIL test/permissionCheck.test.ts
  ● Test suite failed to run

    test/permissionCheck.test.ts:20:43 - error TS2304:
    Cannot find name 'PermissionCheck'.

      20 cdk.Aspects.of(stack).add(new Permi
         ~~~~~

Test Suites: 1 failed, 1 passed, 2 total
Tests: 1 passed, 1 total
Snapshots: 0 total
Time: 4.761 s
Ran all test suites.

Watch Usage: Press w to show more.

aws summit>git checkout asp0
Switched to branch 'asp0'
aws summit>git checkout asp1
Switched to branch 'asp1'
aws summit>
```

We can make this test better as below

```
TS permissionCheck.test.ts test/permissionCheck.test.ts/...
import * as cdk from '@aws-cdk/core';
import '@aws-cdk/assert/jest';
import * as iam from '@aws-cdk/aws-iam';
import { SynthUtils } from '@aws-cdk/assert';
import { Effect } from '@aws-cdk/aws-iam';

class PermissionCheck implements cdk.IAspect {
  visit(node: cdk.IConstruct): void {
    cdk.Annotations.of(node).addWarning('Wildcard resources aren\'t allowed');
  }
}

describe('Given that scope is provided', () => {
  describe('When there is an allow policy', () => {
    test('and no resource restrictions then warning is added', () => {
      const stack = new cdk.Stack();

      const role = new iam.Role(stack, 'myrole.iamrole', {
        assumedBy: new iam.ServicePrincipal('sns.amazonaws.com'),
      });

      role.addToPolicy(new iam.PolicyStatement({
        effect: Effect.ALLOW,
        resources: ['*'],
        actions: ['lambda:InvokeFunction'],
      }));

      cdk.Aspects.of(stack).add(new PermissionCheck());

      expect(stack).toHaveWarning('Wildcard resources aren\'t allowed');
    });
  });
});

declare global {
  namespace jest {
    interface Matchers<R> {
      toHaveWarning(message: string): R;
    }
  }
}

TERMINAL 1: pwsh, pwsh
RUNS test/tdd-myklingon.test.ts
RUNS test/permissionCheck.test.ts

Test Suites: 0 of 2 total
Tests: 0 total
Snapshots: 0 total
Time: 3 s

aws summit>git checkout asp0
Switched to branch 'asp0'
aws summit>git checkout asp1
Switched to branch 'asp1'
aws summit>git checkout asp3
Switched to branch 'asp3'
aws summit>
```

We can see next test


```
TS permissionCheck.test.ts test/permissionCheck.test.ts/...
import * as cdk from '@aws-cdk/core';
import '@aws-cdk/assert/jest';
import * as iam from '@aws-cdk/aws-iam';
import { SynthUtils } from '@aws-cdk/assert';
import { Effect } from '@aws-cdk/aws-iam';

class PermissionCheck implements cdk.IAspect {
  visit(node: cdk.IConstruct): void {
    cdk.Annotations.of(node).addWarning('Wildcard resources aren\'t allowed');
  }
}

describe('Given that scope is provided', () => {
  describe('When there is an allow policy', () => {
    test('and no resource restrictions then warning is added', () => {
      const stack = new cdk.Stack();

      const role = new iam.Role(stack, 'myrole.iamrole', {
        assumedBy: new iam.ServicePrincipal('sns.amazonaws.com'),
      });

      role.addToPolicy(new iam.PolicyStatement({
        effect: Effect.ALLOW,
        resources: ['*'],
        actions: ['lambda:InvokeFunction'],
      }));

      cdk.Aspects.of(stack).add(new PermissionCheck());

      expect(stack).toHaveWarning('Wildcard resources aren\'t allowed');
    });

    test('and resource specified then not warning is added', () => {
      const stack = new cdk.Stack();
      const role = new iam.Role(stack, 'myrole.iamrole', {
        assumedBy: new iam.ServicePrincipal('sns.amazonaws.com'),
      });

      role.addToPolicy(new iam.PolicyStatement({
        effect: Effect.ALLOW,
        resources: ['*'],
        actions: ['lambda:InvokeFunction'],
      }));

      cdk.Aspects.of(stack).add(new PermissionCheck());

      expect(stack).toHaveWarning('Wildcard resources aren\'t allowed');
    });
  });
});
```

TERMINAL 1: pwsh, pwsh

RUNS test/permissionCheck.test.ts
RUNS test/tdd-myklingon.test.ts

Test Suites: 0 of 2 total
Tests: 0 total
Snapshots: 0 total
Time: 0 s, estimated 3 s

aws summit>git checkout asp0
Switched to branch 'asp0'
aws summit>git checkout asp1
Switched to branch 'asp1'
aws summit>git checkout asp3
Switched to branch 'asp3'
aws summit>git checkout asp4
Switched to branch 'asp4'
aws summit>

```
TS permissionCheck.test.ts test/permissionCheck.test.ts/...
const stack = new cdk.Stack();

const role = new iam.Role(stack, 'myrole.iamrole', {
  assumedBy: new iam.ServicePrincipal('sns.amazonaws.com'),
});

role.addToPolicy(new iam.PolicyStatement({
  effect: Effect.ALLOW,
  resources: ['*'],
  actions: ['lambda:InvokeFunction'],
}));

cdk.Aspects.of(stack).add(new PermissionCheck());

expect(stack).toHaveWarning('Wildcard resources aren\'t allowed');
});

test('and resource specified then not warning is added', () => {
  const stack = new cdk.Stack();
  const role = new iam.Role(stack, 'myrole.iamrole', {
    assumedBy: new iam.ServicePrincipal('sns.amazonaws.com'),
  });

  role.addToPolicy(new iam.PolicyStatement({
    effect: Effect.ALLOW,
    resources: ['mylambda'],
    actions: ['lambda:InvokeFunction'],
  }));

  cdk.Aspects.of(stack).add(new PermissionCheck());

  expect(stack).not.toHaveWarning('Wildcard resources aren\'t allowed');
});
});
```

TERMINAL 1: pwsh, pwsh

PASS test/tdd-myklingon.test.ts
FAIL test/permissionCheck.test.ts

• Given that scope is provided > When there is an allow policy > and resource specified then not warning is added

Template has expected error

```
45 | cdk.Aspects.of(stack).add(new P
46 | ermissionCheck());
47 | expect(stack).not.toHaveWarning
48 | ('Wildcard resources aren\'t allowed');
49 |
50 | });
```

at Object.<anonymous> (test/permissionCheck.test.ts:47:31)

Test Suites: 1 failed, 1 passed, 2 total
Tests: 1 failed, 2 passed, 3 total
Snapshots: 0 total
Time: 4.264 s
Ran all test suites.

Watch Usage: Press w to show more.

aws summit>git checkout asp0
Switched to branch 'asp0'
aws summit>git checkout asp1
Switched to branch 'asp1'
aws summit>git checkout asp3
Switched to branch 'asp3'
aws summit>git checkout asp4
Switched to branch 'asp4'
aws summit>git checkout asp5

Next test again

```
TS permissionCheck.test.ts test/permissionCheck.test.ts/...
import '@aws-cdk/assert/jest';
import * as iam from '@aws-cdk/aws-iam';
import { SynthUtils } from '@aws-cdk/assert';
import { Effect } from '@aws-cdk/aws-iam';

class PermissionCheck implements cdk.IAspect {
  visit(node: cdk.IConstruct): void {
    if (node instanceof iam.CfnPolicy) {
      var hasWildcardResource = node.policyDocument.statements
        .some((x: any) => x.resource.some((r: any) => r == '*'));

      if (hasWildcardResource) {
        cdk.Annotations.of(node).addWarning('Wildcard resources aren\'t allowed')
      }
    }
  }
}

describe('Given that scope is provided', () => {
  describe('When there is an allow policy', () => {
    test('and no resource restrictions then warning is added', () => {
      const stack = new cdk.Stack();

      const role = new iam.Role(stack, 'myrole.iamrole', {
        assumedBy: new iam.ServicePrincipal('sns.amazonaws.com'),
      });

      role.addToPolicy(new iam.PolicyStatement({
        effect: Effect.ALLOW,
        resources: ['*'],
        actions: ['lambda:InvokeFunction'],
      }));

      cdk.Aspects.of(stack).add(new PermissionCheck());

      expect(stack).toHaveWarning('Wildcard resources aren\'t allowed');
    });
  });
});
```

TERMINAL ... 1: pwsh, pwsh + - - -

PASS test/tdd-myklingon.test.ts
PASS test/permissionCheck.test.ts

Test Suites: 2 passed, 2 total
Tests: 3 passed, 3 total
Snapshots: 0 total
Time: 3.754 s
Ran all test suites.

Watch Usage: Press w to show more.

aws summit>git checkout asp0
Switched to branch 'asp0'
aws summit>git checkout asp1
Switched to branch 'asp1'
aws summit>git checkout asp3
Switched to branch 'asp3'
aws summit>git checkout asp4
Switched to branch 'asp4'
aws summit>git checkout asp5
Switched to branch 'asp5'
aws summit>

Switch again to check for deny

```
TS permissionCheck.test.ts test/permissionCheck.test.ts/...
import '@aws-cdk/assert/jest';
import * as iam from '@aws-cdk/aws-iam';
import { SynthUtils } from '@aws-cdk/assert';
import { Effect } from '@aws-cdk/aws-iam';

class PermissionCheck implements cdk.IAspect {
  visit(node: cdk.IConstruct): void {
    if (node instanceof iam.CfnPolicy) {
      var hasWildcardResource = node.policyDocument.statements
        .some((x: any) => x.resource.some((r: any) => r == '*'));

      if (hasWildcardResource) {
        cdk.Annotations.of(node).addWarning('Wildcard resources aren\'t allowed')
      }
    }
  }
}

describe('Given that scope is provided', () => {
  test('When there is a deny policy wildcard resource produce no warning', () => {
    const stack = new cdk.Stack();

    const role = new iam.Role(stack, 'myrole.iamrole', {
      assumedBy: new iam.ServicePrincipal('sns.amazonaws.com'),
    });

    role.addToPolicy(new iam.PolicyStatement({
      effect: Effect.DENY,
      resources: ['*'],
      actions: ['lambda:InvokeFunction'],
    }));

    cdk.Aspects.of(stack).add(new PermissionCheck());

    expect(stack).not.toHaveWarning('Wildcard resources aren\'t allowed');
  });

  describe('When there is an allow policy', () => {
    test('and no resource restrictions then warning is added', () => {
      // ... (previous test code)
    });
  });
});
```

TERMINAL ... 1: pwsh, pwsh + - - -

PASS test/tdd-myklingon.test.ts
FAIL test/permissionCheck.test.ts

• Given that scope is provided › When there is a deny policy wildcard resource produce no warning

Template has expected error

```
34 | cdk.Aspects.of(stack).add(new PermissionCheck());
35 |
36 | expect(stack).not.toHaveWarning('Wildcard resources aren\'t allowed');
37 | });
38 | describe('When there is an allow policy', () => {
39 |   test('and no resource restrictions then warning is added', () => {
    at Object.<anonymous> (test/permissionCheck.test.ts:36:27)
```

Test Suites: 1 failed, 1 passed, 2 total
Tests: 1 failed, 3 passed, 4 total
Snapshots: 0 total
Time: 3.22 s
Ran all test suites.

Watch Usage: Press w to show more.

aws summit>git checkout asp0
Switched to branch 'asp0'
aws summit>git checkout asp1
Switched to branch 'asp1'
aws summit>git checkout asp3
Switched to branch 'asp3'
aws summit>git checkout asp4
Switched to branch 'asp4'
aws summit>git checkout asp5
Switched to branch 'asp5'
aws summit>git checkout asp5.5
Switched to branch 'asp5.5'
aws summit>

Now to start fixing the tests

TS permissionCheck.test.ts test/permissionCheck.test.ts/...

```
import '@aws-cdk/assert/jest';
import * as iam from '@aws-cdk/aws-iam';
import { SynthUtils } from '@aws-cdk/assert';
import { Effect } from '@aws-cdk/aws-iam';

class PermissionCheck implements cdk.IAspect {
  visit(node: cdk.IConstruct): void {
    if (node instanceof iam.CfnPolicy) {
      var hasWildcardResource = node.policyDocument.statements
        .some((x: any) => x.effect == 'Allow'
          && x.resource.some((r: any) => r == '*'));

      if (hasWildcardResource) {
        cdk.Annotations.of(node).addWarning('Wildcard resources aren\'t allowed')
      }
    }
  }
}

describe('Given that scope is provided', () => {
  test('When there is a deny policy wildcard resource produce no warning', () => {
    const stack = new cdk.Stack();

    const role = new iam.Role(stack, 'myrole.iamrole', {
      assumedBy: new iam.ServicePrincipal('sns.amazonaws.com'),
    });

    role.addToPolicy(new iam.PolicyStatement({
      effect: Effect.DENY,
      resources: ['*'],
      actions: ['lambda:InvokeFunction'],
    }));

    cdk.Aspects.of(stack).add(new PermissionCheck());

    expect(stack).not.toHaveWarning('Wildcard resources aren\'t allowed');
  });
});
```

TERMINAL 1: pwsh, pwsh

PASS test/tdd-mykington.test.ts

PASS test/permissionCheck.test.ts

Test Suites: 2 passed, 2 total

Tests: 4 passed, 4 total

Snapshots: 0 total

Time: 3.821 s

Ran all test suites.

Watch Usage: Press w to show more.

Switched to branch 'asp0'

aws summit>git checkout asp1

Switched to branch 'asp1'

aws summit>git checkout asp3

Switched to branch 'asp3'

aws summit>git checkout asp4

Switched to branch 'asp4'

aws summit>git checkout asp5

Switched to branch 'asp5'

aws summit>git checkout asp5.5

Switched to branch 'asp5.5'

aws summit>git checkout asp6

Switched to branch 'asp6'

aws summit>

FAQ

Questions	Possible cause and solution
I have too many tests	You probably created tests for the code, not for the requirements. Try to write a test not when you need to add a class but when you have a new functional expectation.
Each time I change something I need to update a large number of tests, I wouldn't call it easy refactoring!	You probably write your tests against implementation and not against interface. Try to think about your implementation like a black box.
It takes too long to code	The initial design is longer, but considering reduced bug density and clear documentation, the delivery is faster.

Resources

AWS CDK Developer Guide

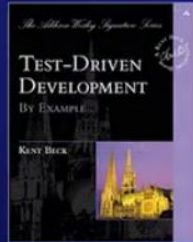
<https://docs.aws.amazon.com/cdk/latest/guide>

Workshop

<https://cdkworkshop.com/>

Realizing quality improvement through test driven development: results and experiences of four industrial teams <https://bit.ly/30m8eYP>

Test Driven Development:
By Example by Kent Beck



© 2021, Amazon Web Services, Inc. or its affiliates. All rights reserved.

What is next?

Practice, practice, and practice

Kata - It is practised in Japanese martial arts as a way to memorise and perfect the movements being executed

Join CDK community <https://cdk.dev/>

Learn DevOps with AWS Training and Certification

RESOURCES CREATED BY THE EXPERTS AT AWS TO HELP YOU BUILD AND VALIDATE DEVOPS SKILLS



Learn online with 45+ free digital courses, including:
Advanced Testing Practices using AWS DevOps Tools (2.5 hours)



Dive deep with classroom training, including:
DevOps Engineering on AWS (3 days)



Build credibility and confidence with AWS Certification, including:
AWS Certified DevOps Engineer – Professional and
AWS Certified Developer – Associate



Go deeper with labs, whitepapers, tech talks, and more by
accessing the [AWS Ramp-Up Guide](#)

Visit aws.training/DevOps