AWS Summit ANZ 2021 - Driving a test-first strategy with CDK and test driven development

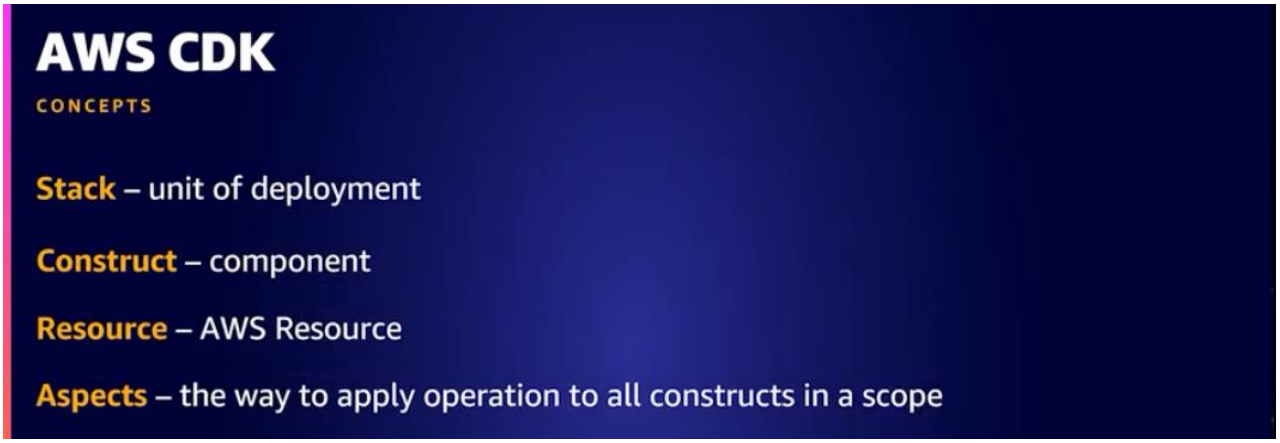
AWS Events
93.6K subscribers

Subscribe

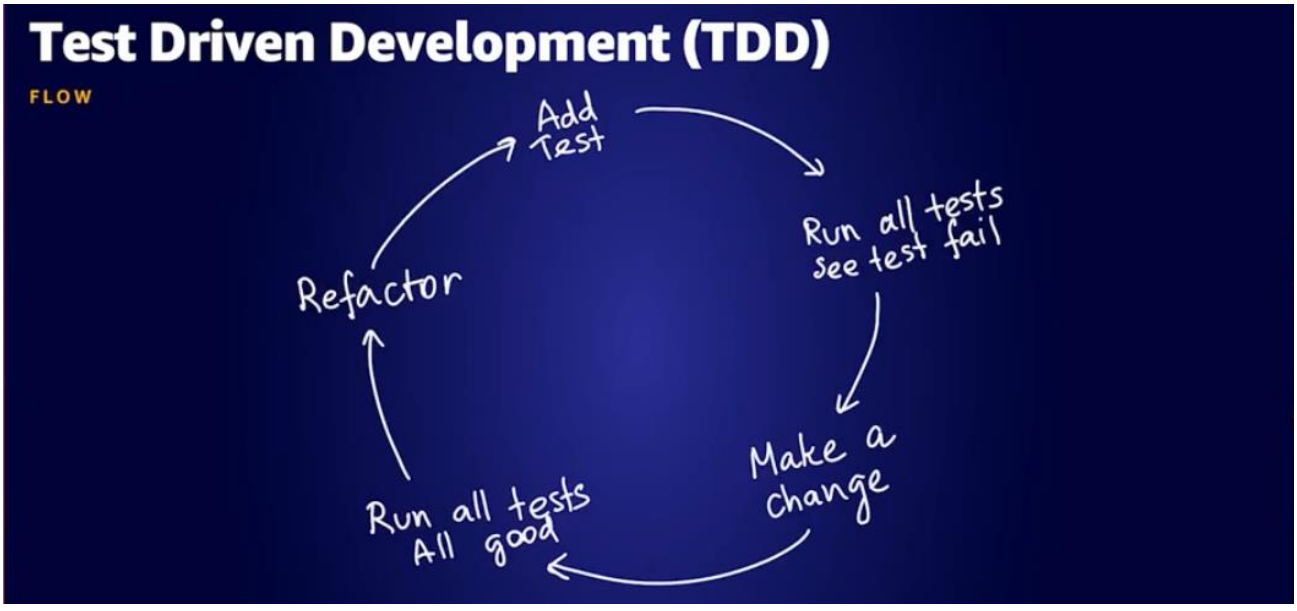
32 | Share | Download | Clip | ...

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.



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.



The story

Demo explained

Problem
Build a communication channel component for others to use.

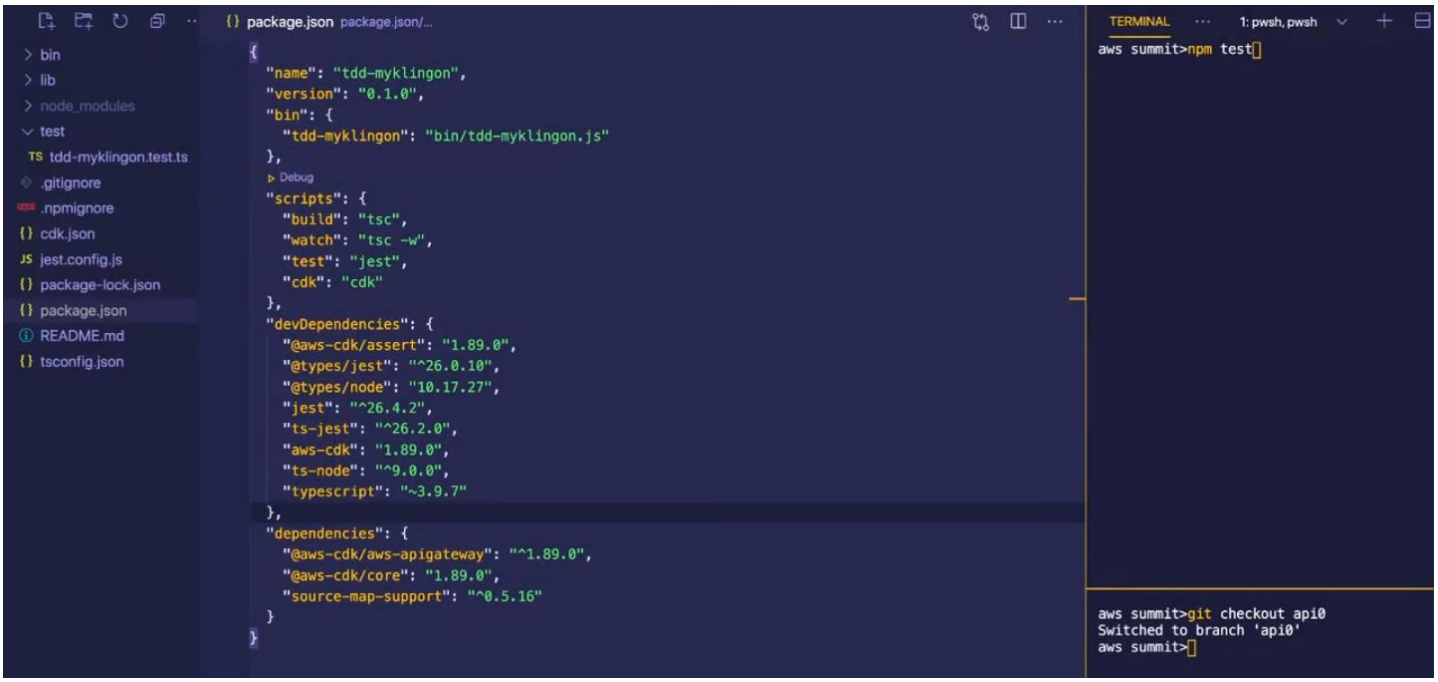
Solution
Split problem into small sub-problems

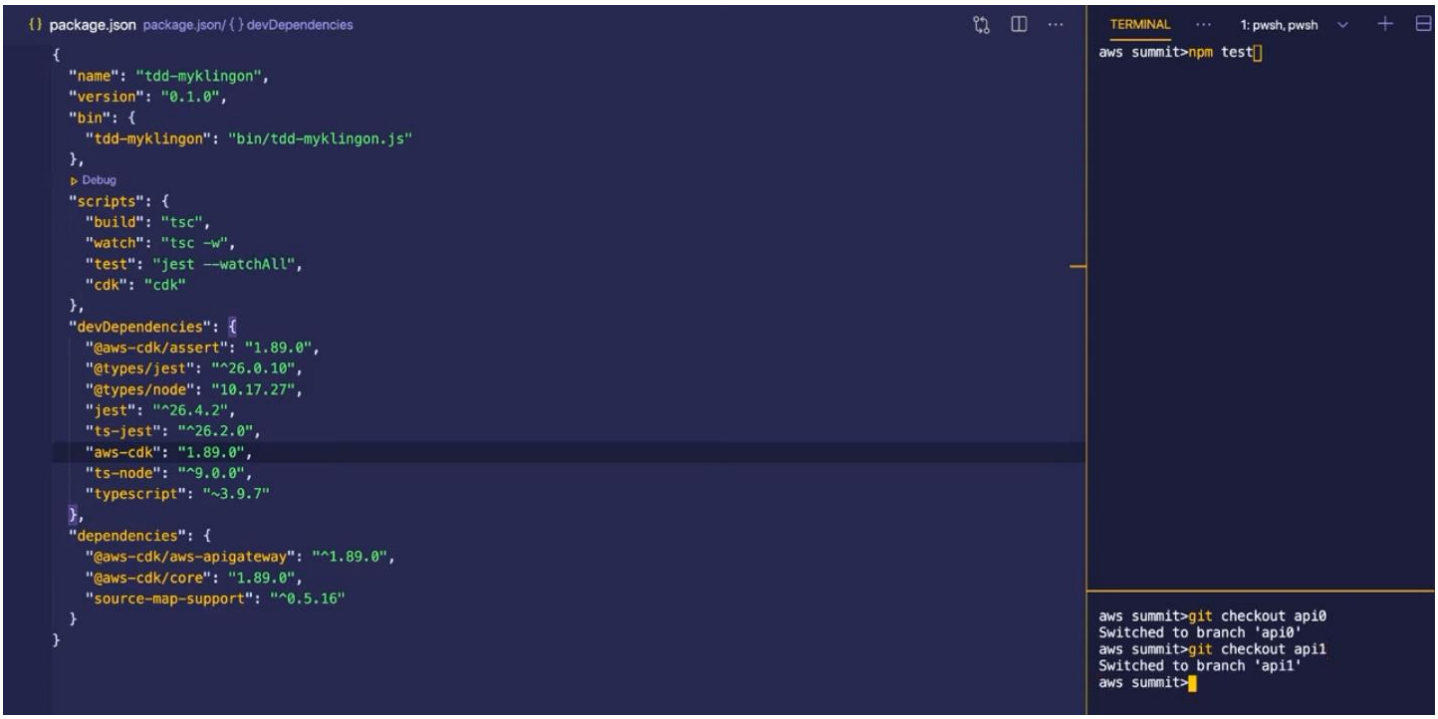
Workflow
Write test for the next sub-problem
Observe it fail
Make it work
Refactor (as needed)
Repeat till the problem is solved

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

Repository https://bit.ly/38ilkd5

Demo
TDD workflow with CDK


bin
lib
node_modules
test
tdd-myklingon.test.ts
.gitignore
.npmignore
cdk.json
jest.config.js
package-lock.json
package.json
README.md
tsconfig.json
package.json package.json/...
{
"name": "tdd-myklingon",
"version": "0.1.0",
"bin": {
"tdd-myklingon": "bin/tdd-myklingon.js"
},
Debug
"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
1: pwsh, pwsh
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"
},
Debug
"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
1: pwsh, pwsh
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

```ts
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'
            })
        });
    });
});
```

TERMINAL 1: pwsh, pwsh

```
aws summit>npm test
```

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

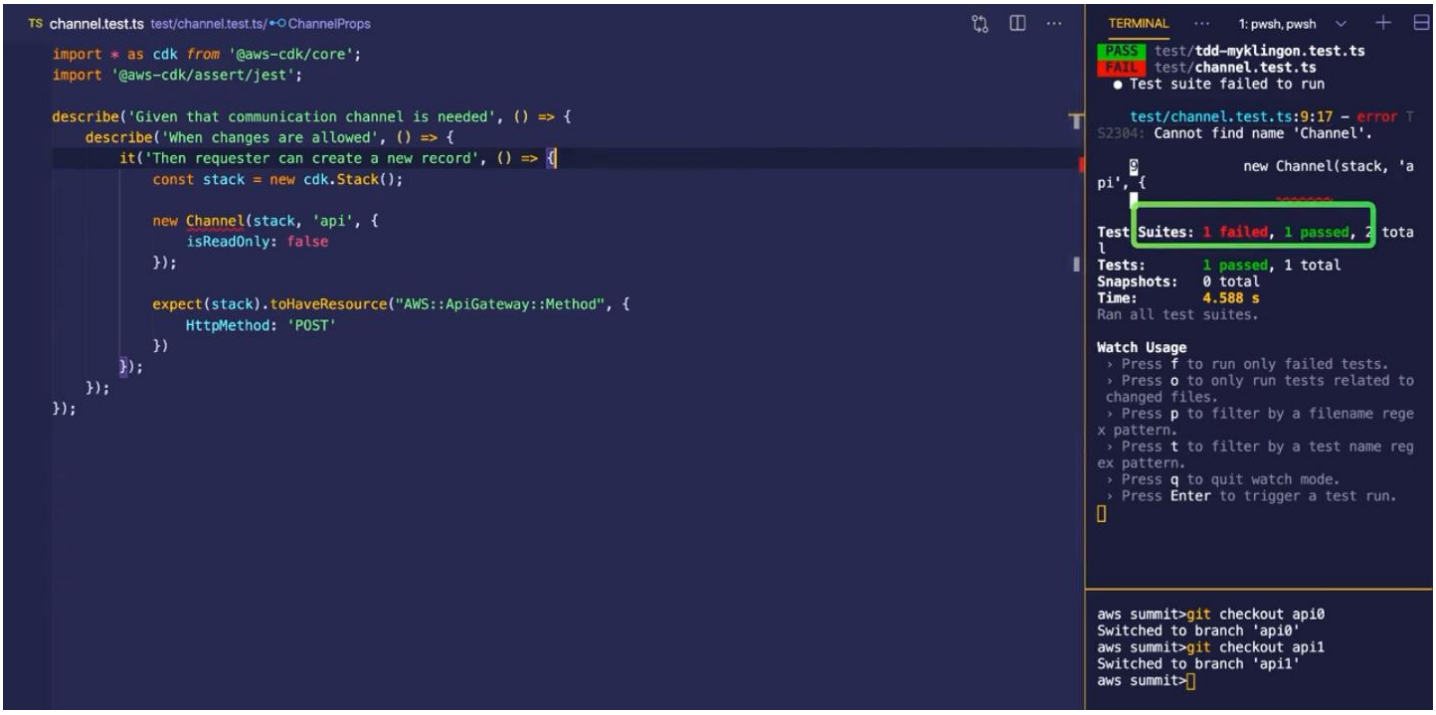
{} package.json package.json/...

```json
{
  "name": "tdd-myklingon",
  "version": "0.1.0",
  "bin": {
    "tdd-myklingon": "bin/tdd-myklingon.js"
  },
  ▷ Debug
  "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"
  }
}
```

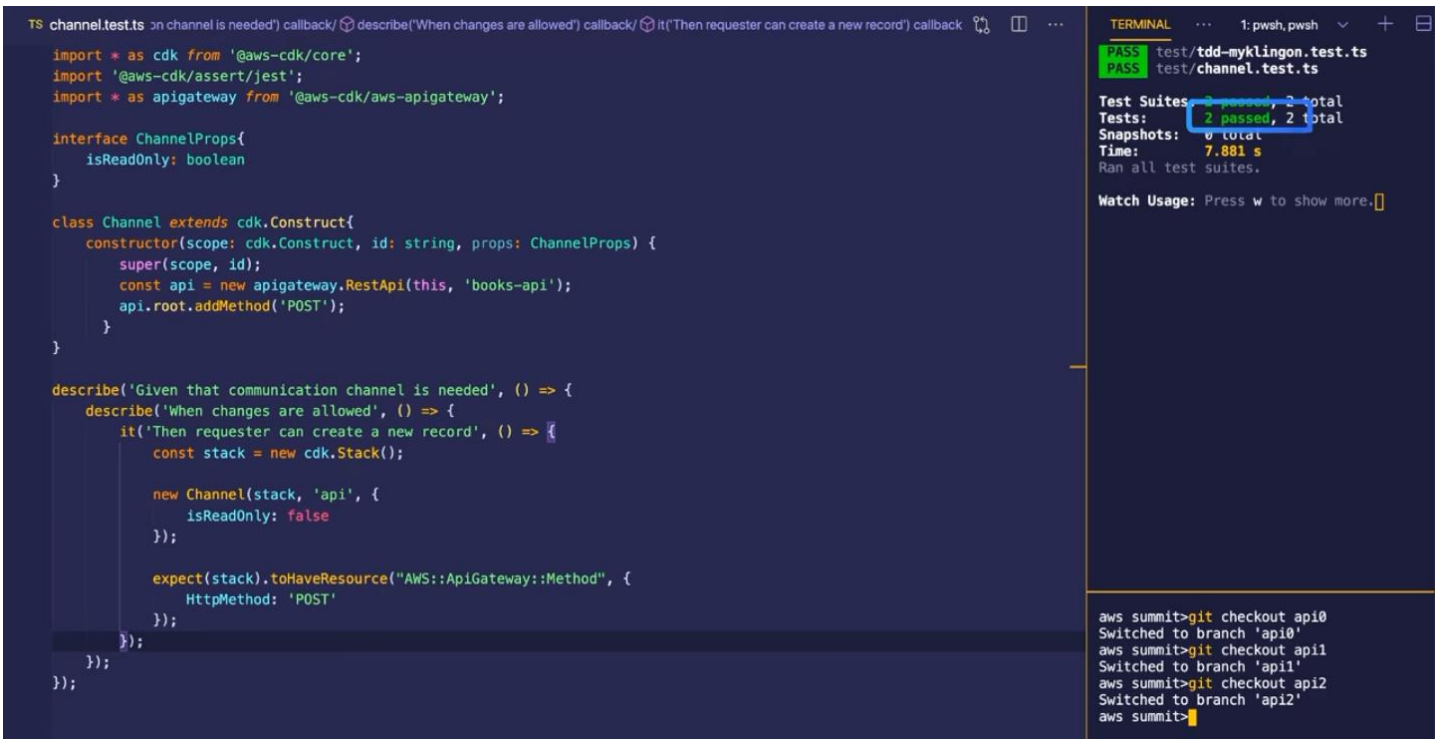
TERMINAL 1: pwsh, pwsh

```
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, 'api', {
                    ~~~~~~~

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.
```

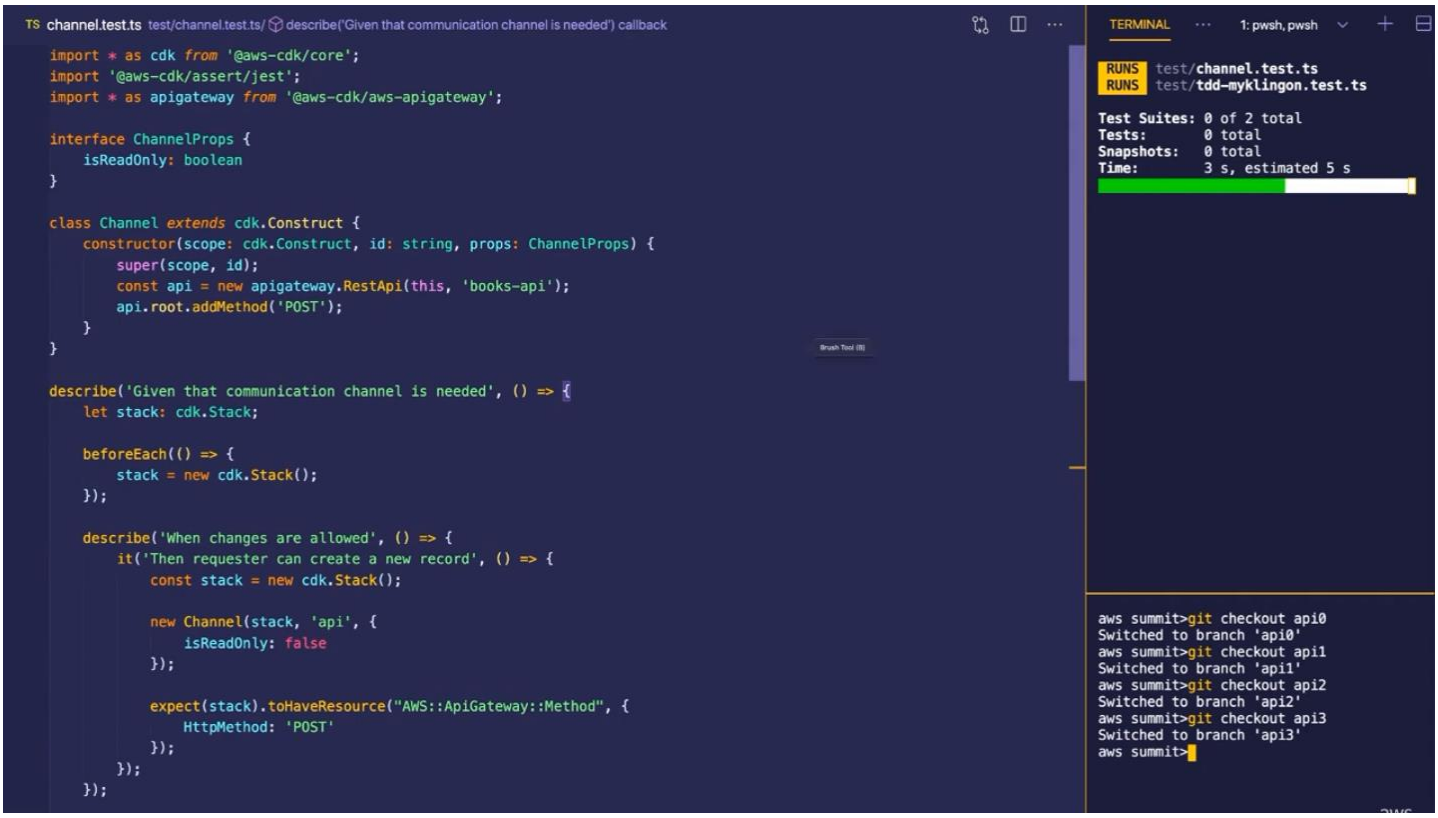
```
aws summit>git checkout api0
Switched to branch 'api0'
aws summit>git checkout api1
Switched to branch 'api1'
aws summit>
```

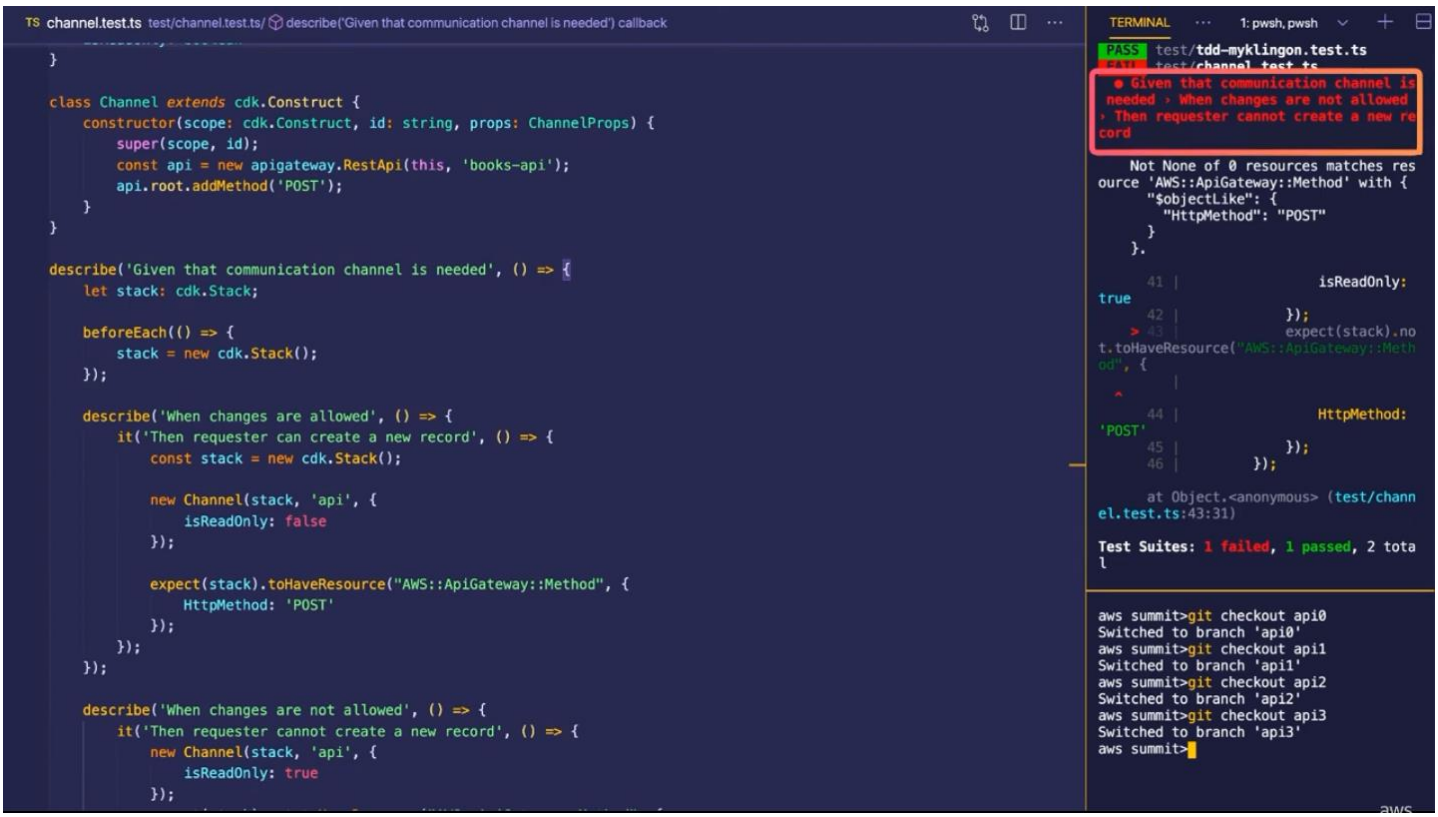


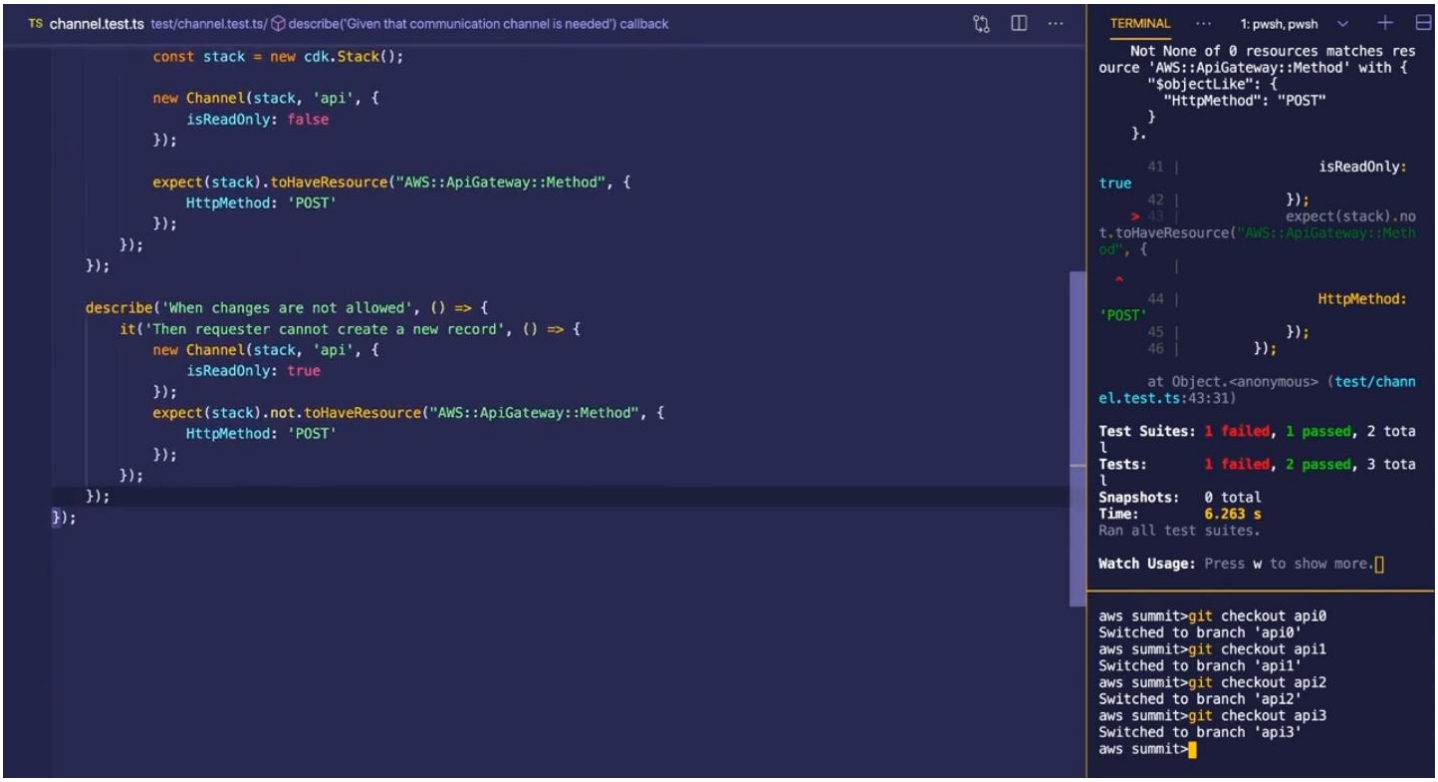
We now need to make this green



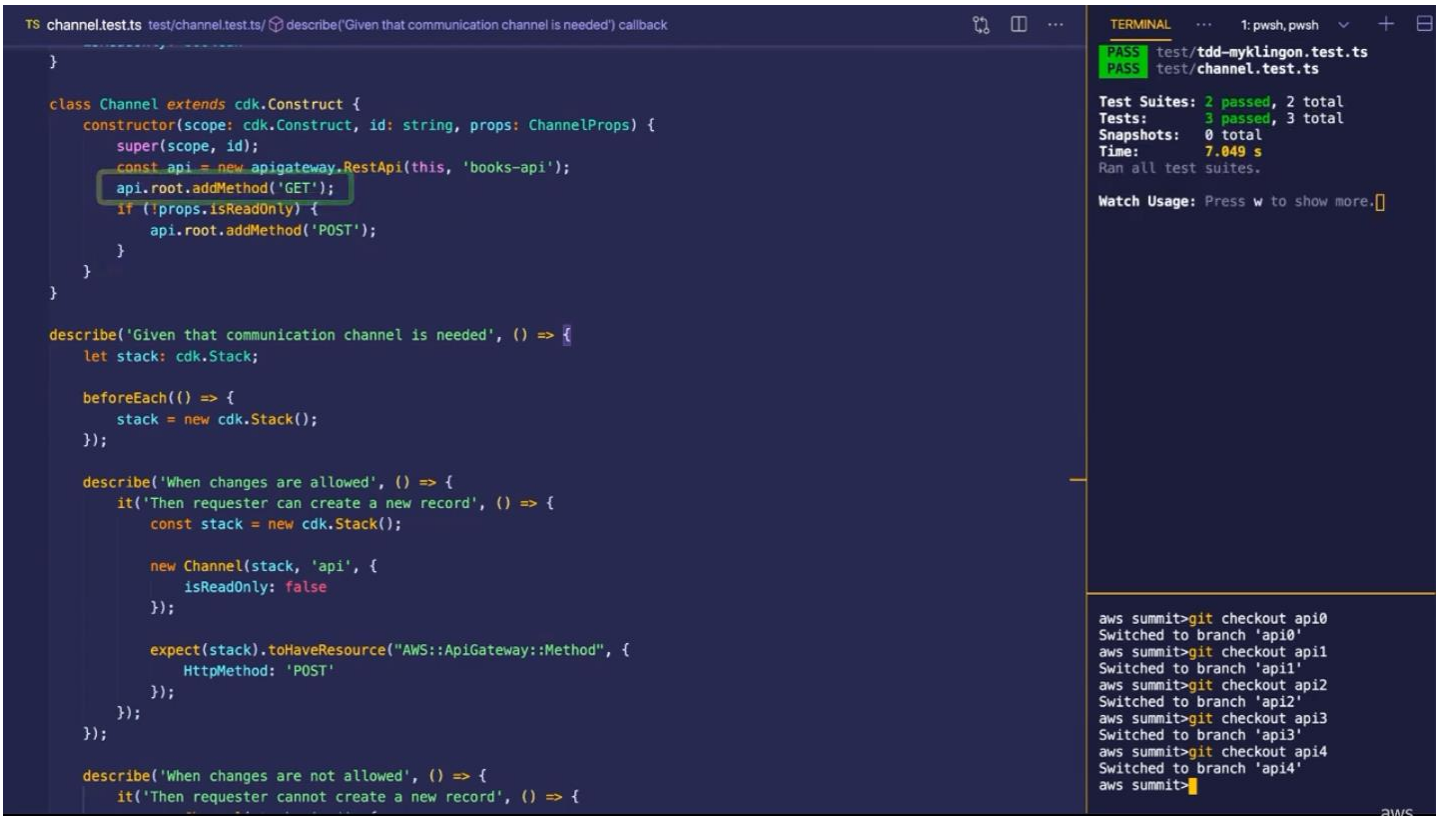
Let’s add more tests

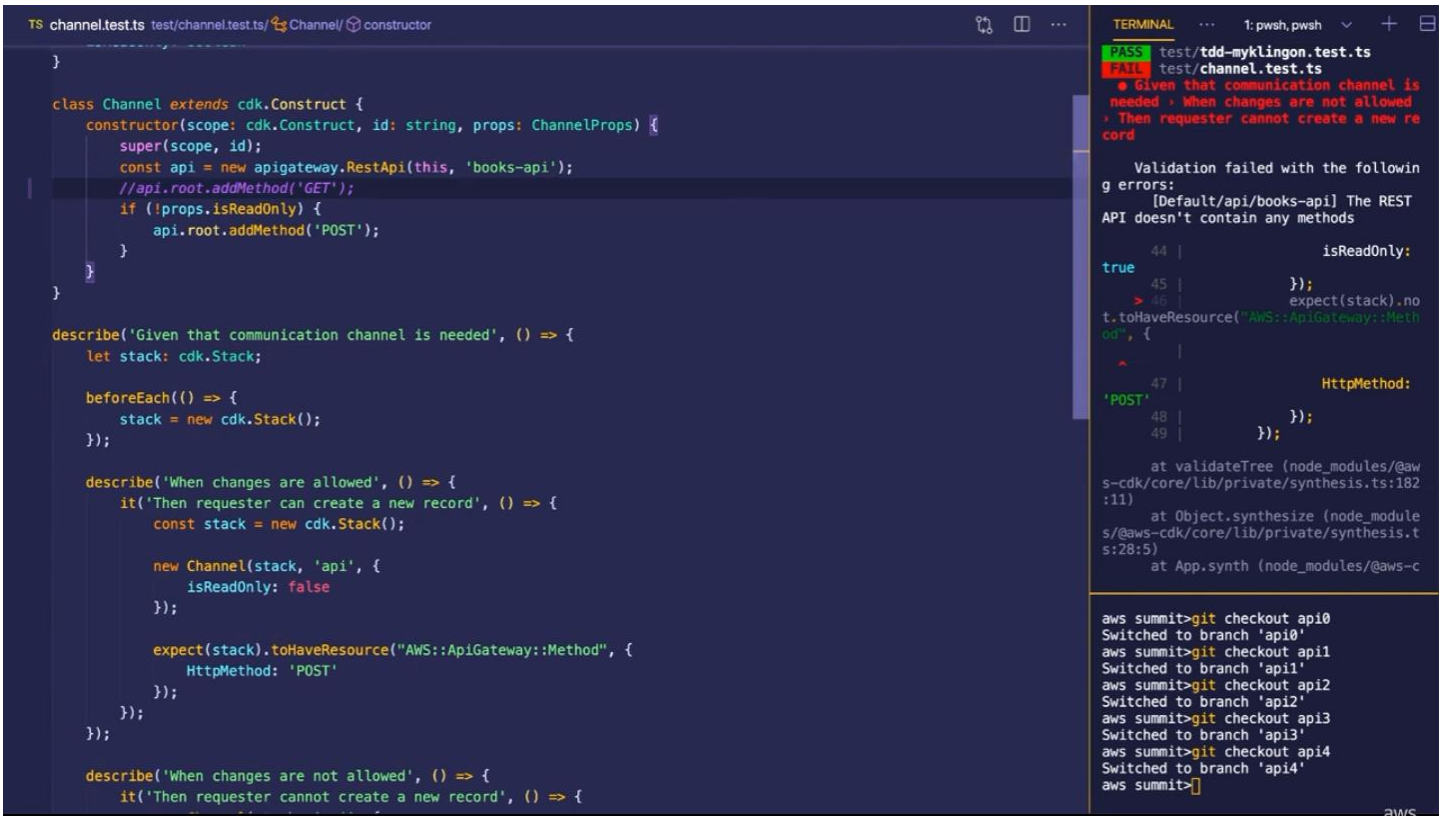

channel.test.ts test/channel.test.ts/ describe('Given that communication channel is needed') callback
import * as cdk from '@aws-cdk/core';
import '@aws-cdk/assert/jest';
import * as apigateway from '@aws-cdk/aws-apigateway';
interface ChannelProps {
isReadOnly: boolean
}
class Channel extends cdk.Construct {
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'
});
});
});
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>


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
● 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
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>

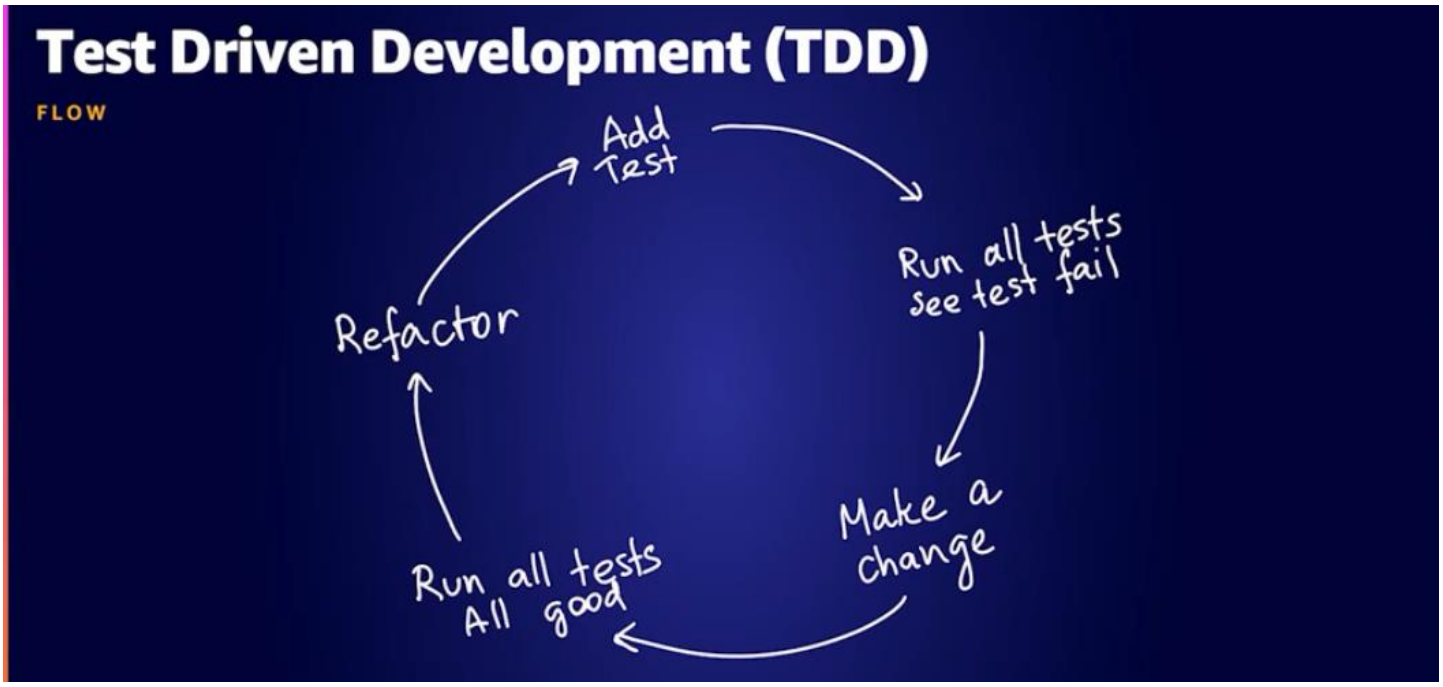


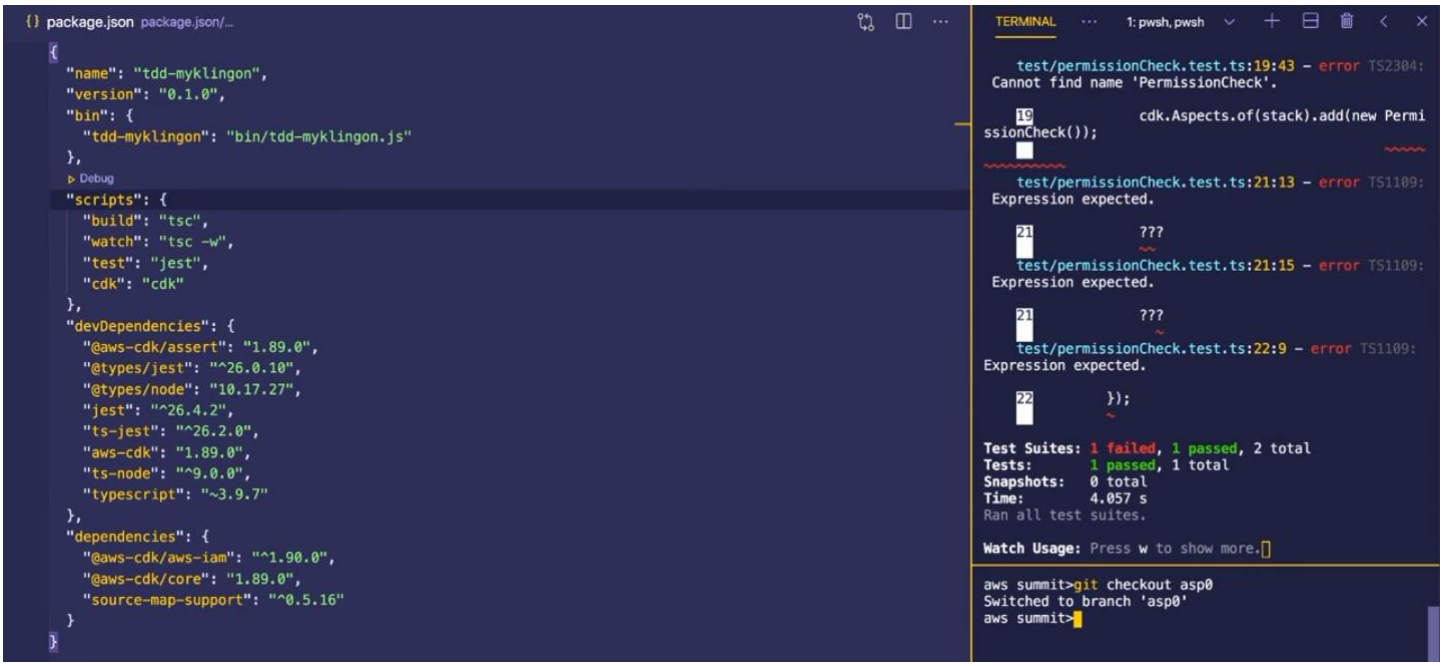
Now let’s fix the error



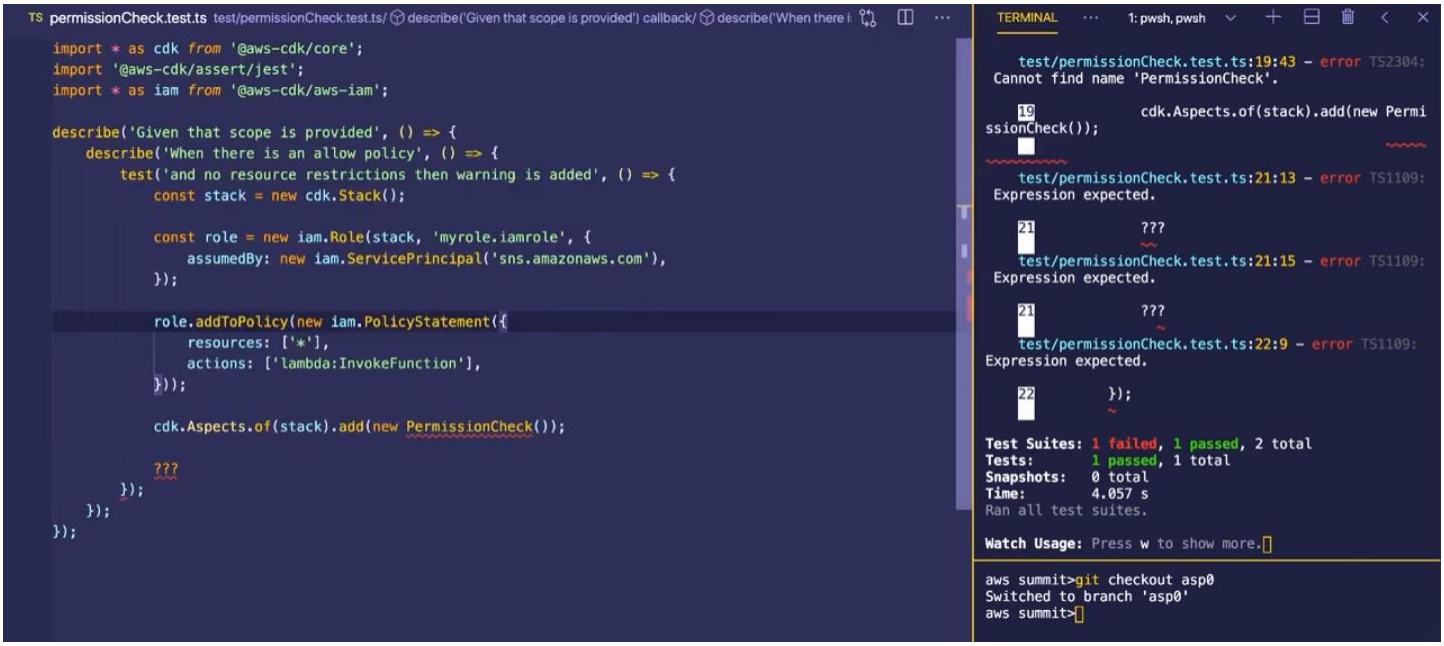


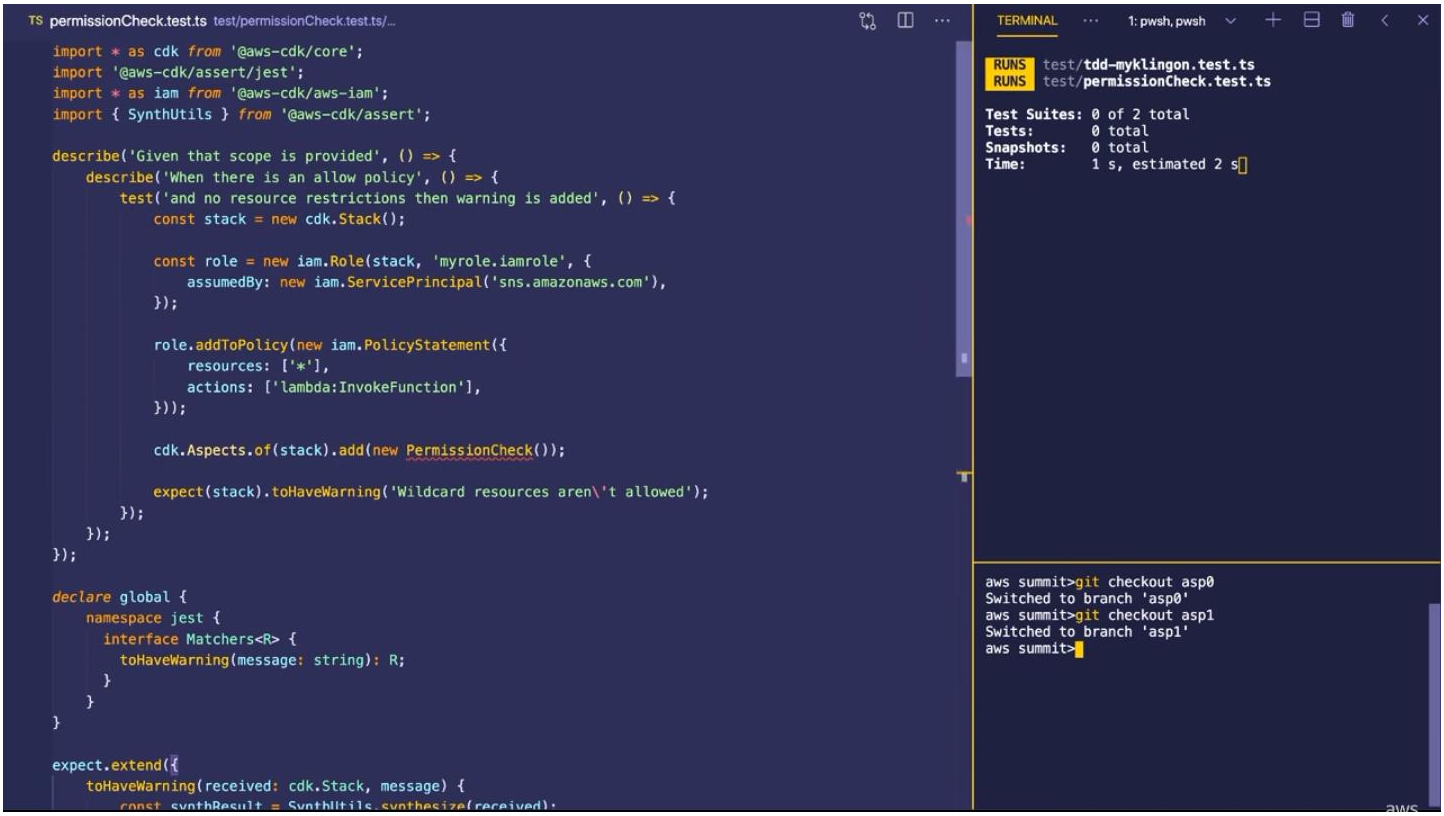
The test fails if we comment out the addMethod as above

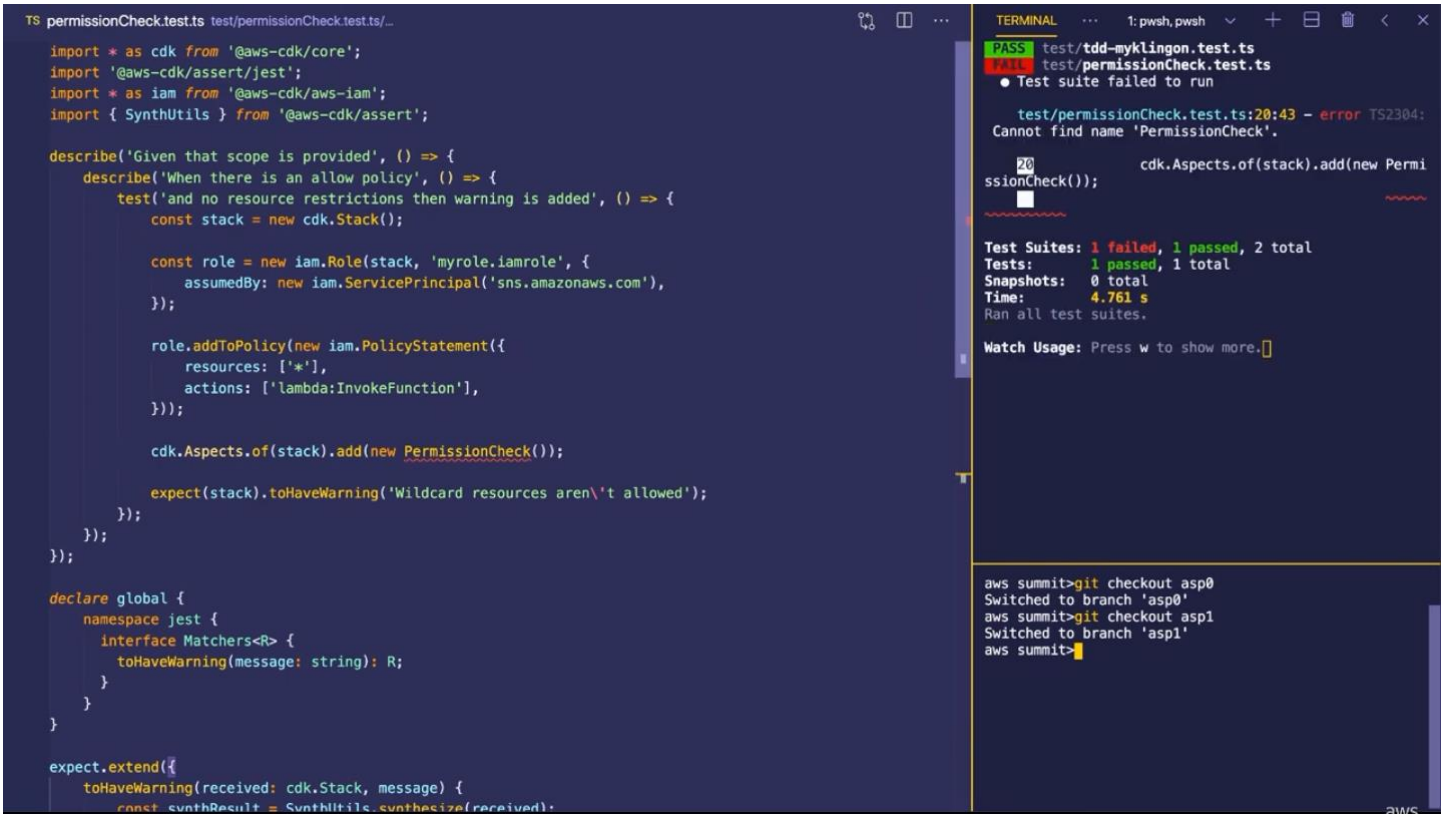


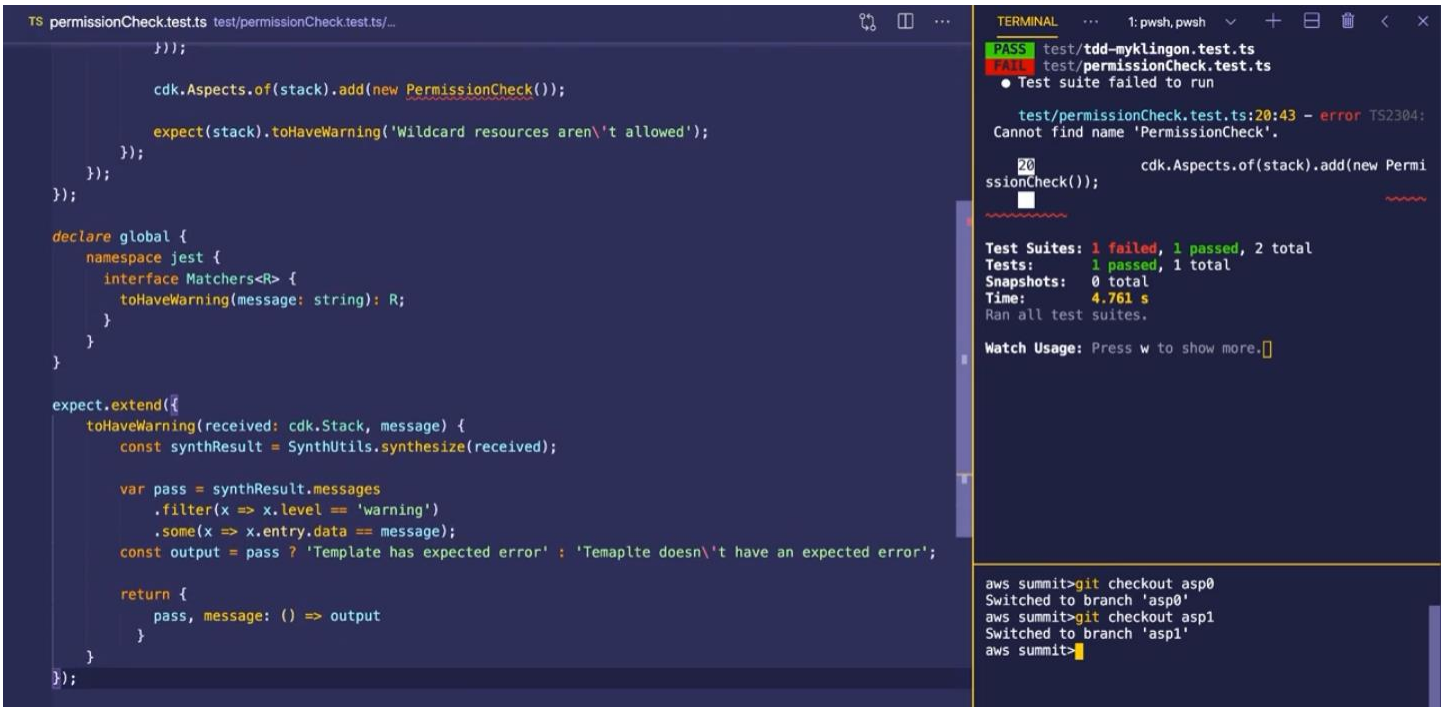


We now have an IAM dependency added and we are using Aspects as below

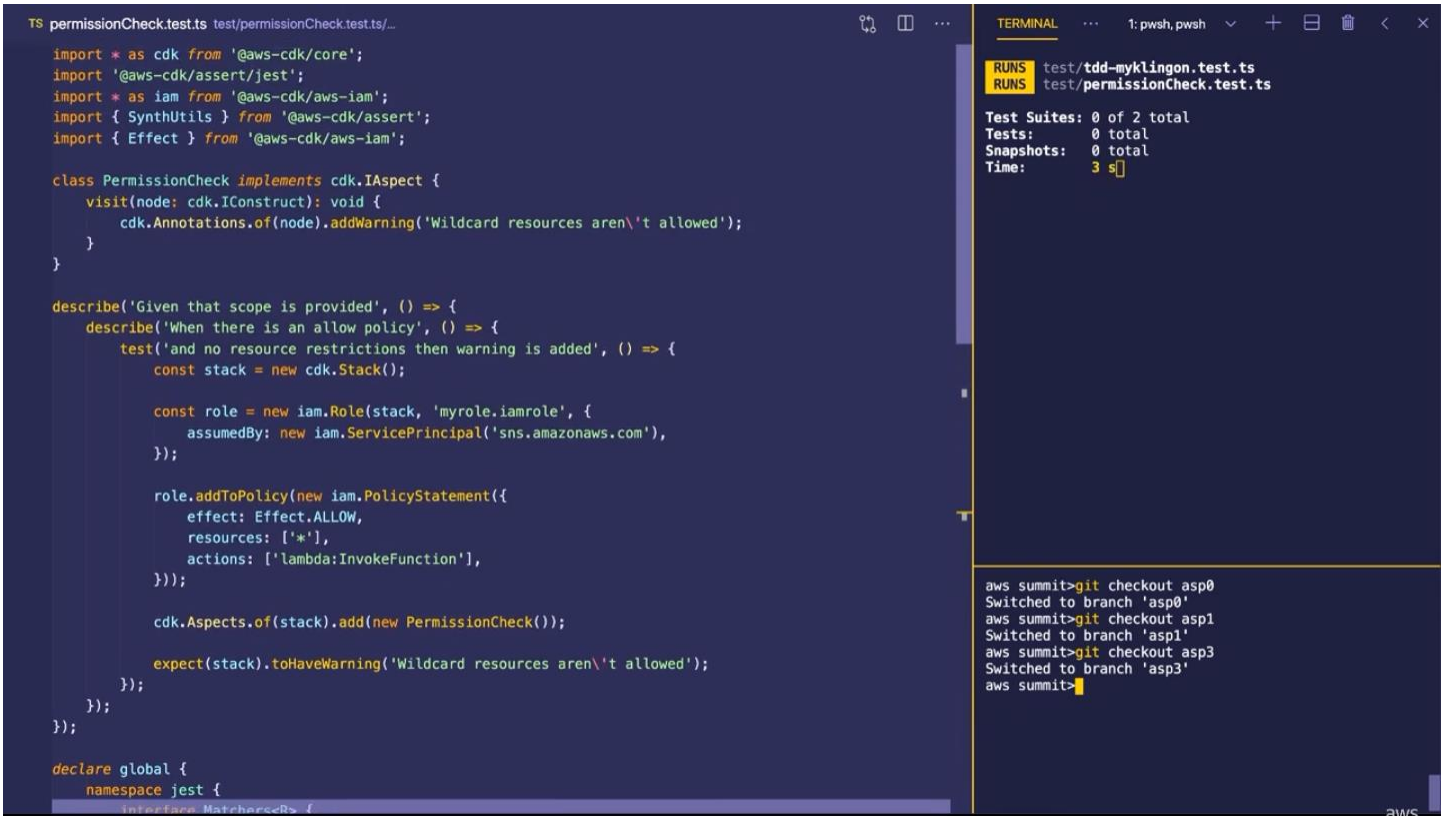



TS permissionCheck.test.ts test/permissionCheck.test.ts/...
import * as cdk from '@aws-cdk/core';
import '@aws-cdk/assert/jest';
import * as iam from '@aws-cdk/aws-iam';
import { SynthUtils } from '@aws-cdk/assert';
describe('Given that scope is provided', () => {
describe('When there is an allow policy',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

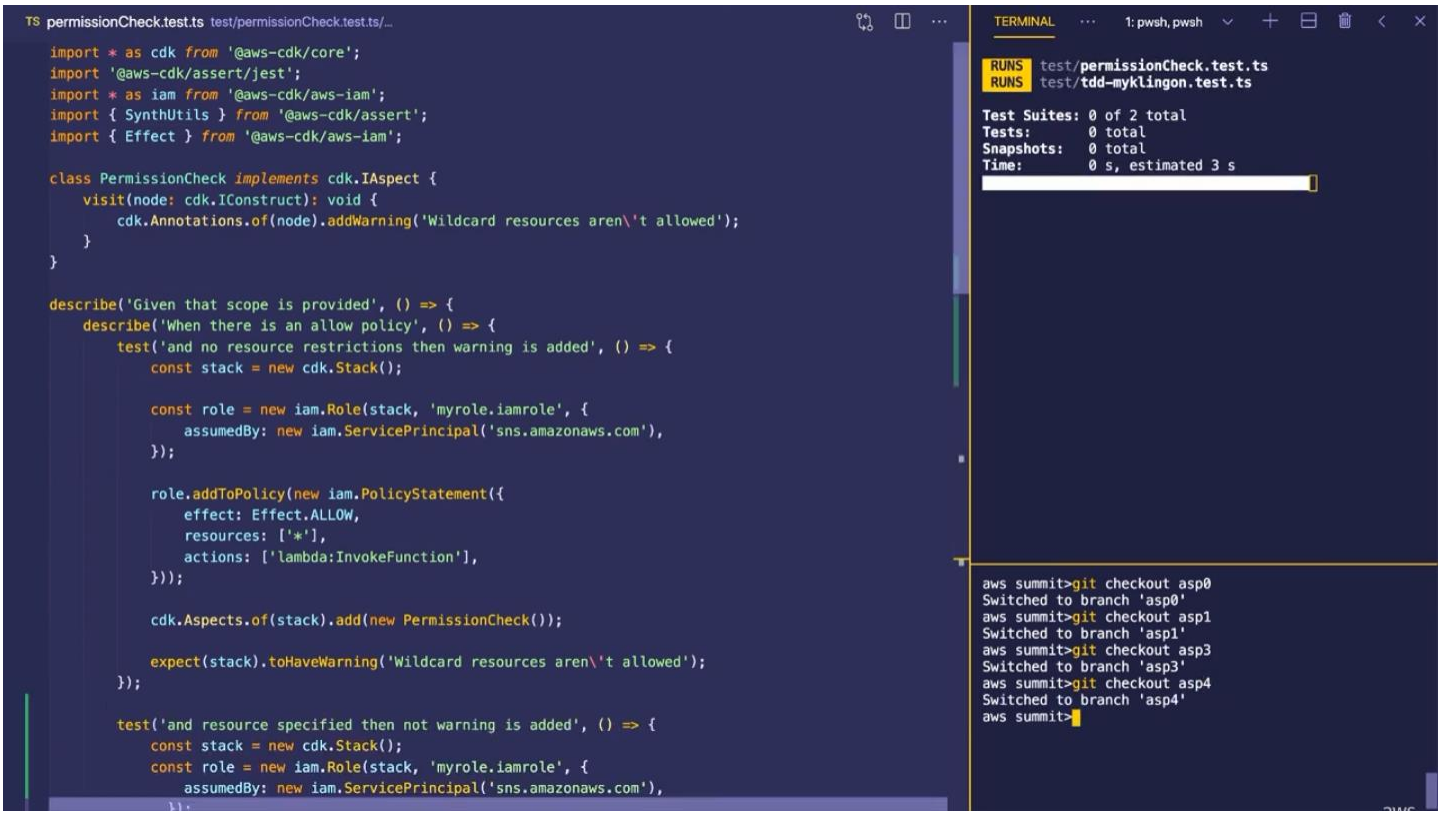

TS permissionCheck.test.ts test/permissionCheck.test.ts/...
import * as cdk from '@aws-cdk/core';
import '@aws-cdk/assert/jest';
import * as iam from '@aws-cdk/aws-iam';
import { SynthUtils } from '@aws-cdk/assert';
describe('Given that scope is provided', () => {
describe('When there is an allow policy',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

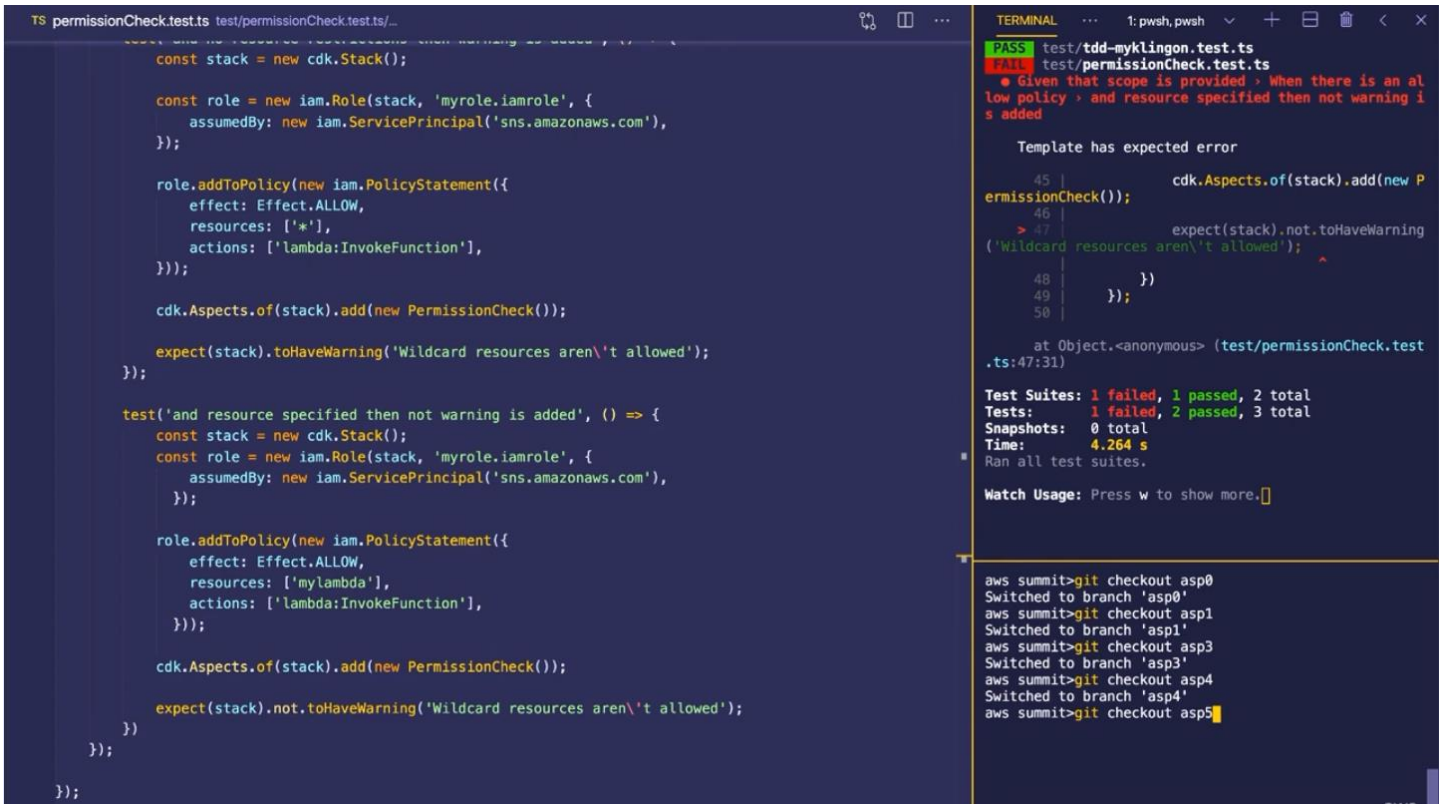


We can make this test better as below

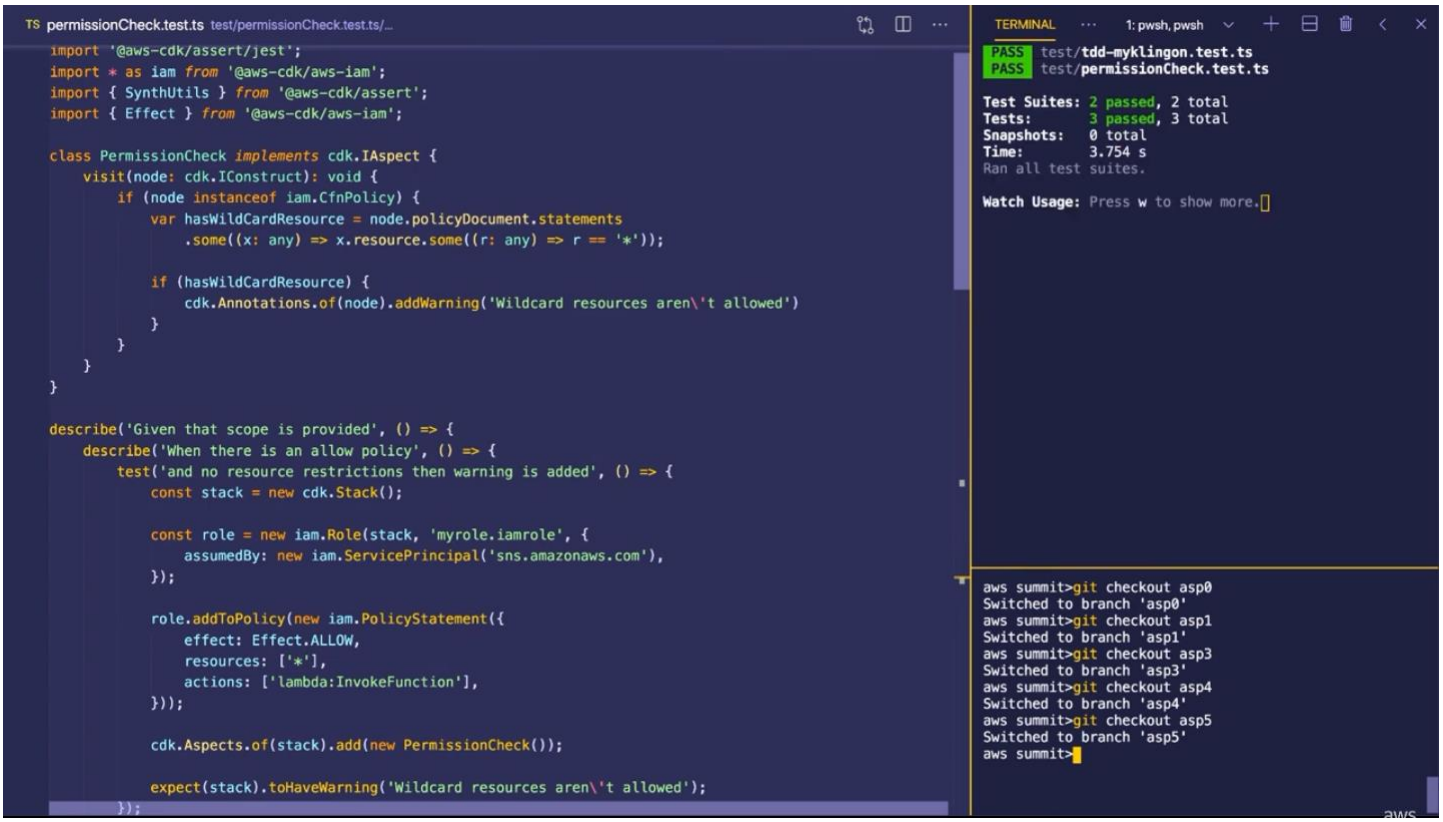


We can see next test

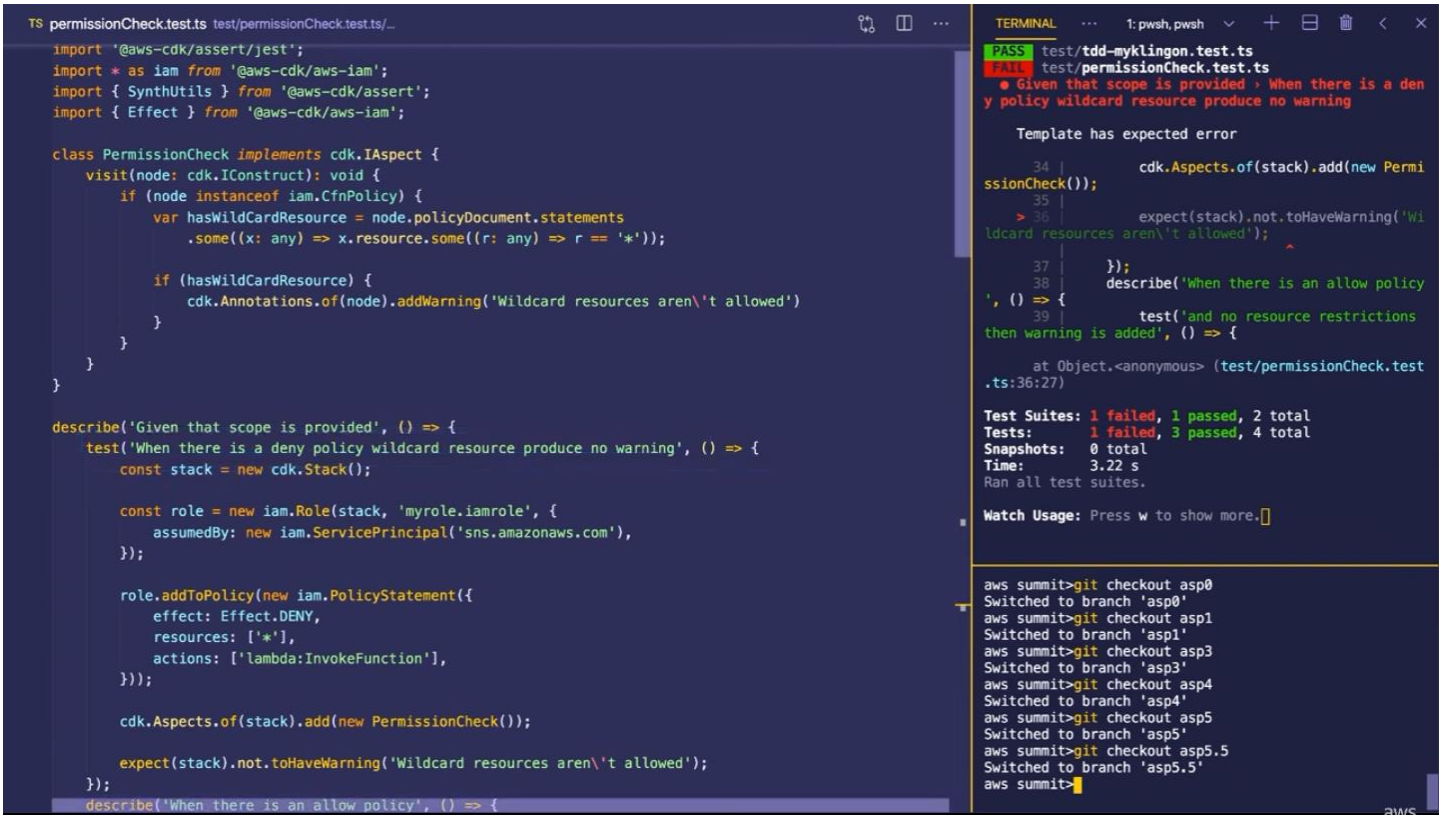




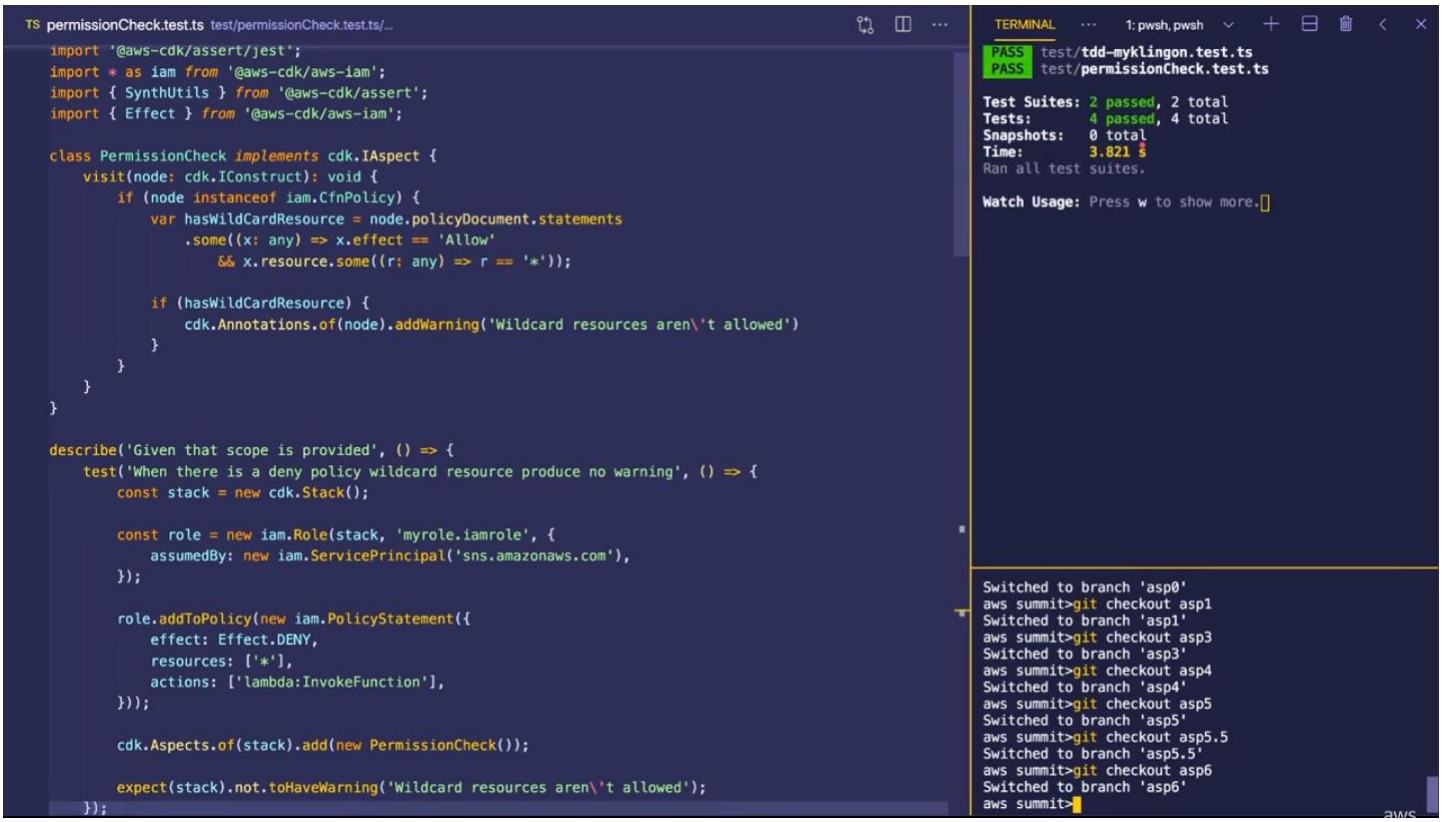
Next test again



Switch again to check for deny



Now to start fixing the tests



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.

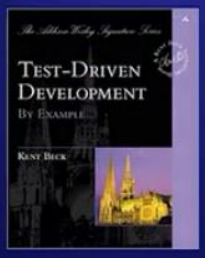
Resources

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



aws

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**