

B-06

AWS CDKにコントリビュートするための 具体的な方法

佐藤智樹

クラスメソッド 株式会社

20.10.2020

自己紹介

- 佐藤智樹
- クラスメソッド株式会社
- CX事業本部（IoTチームと自社サービス開発）
- 前職：SIerのSEを4年弱
- 現在：サーバーサイドエンジニア
- 好きなAWSサービス：Lambda、CDK



 @tmk2154

 @tomoki10



アジェンダ

- 対象とする視聴者
- コントリビュートすると良いこと
- AWS CDKの構成と詳細
- コントリビューティングガイドの説明
- 実際にコントリビュートした流れ
- まとめ



対象とする視聴者

- AWS CDK or 大きめのOSSにコントリビュートしてみたい人
エンジニアやってるんだし一度ぐらいはやってみたい
せっかくだしドキュメントだけでなくソースもコミットしたい
- AWS CDKは内部構成を少し知りたい
適宜説明は入れますが、AWS CDKのBlackBeltを見ていると分かりやすいかも

目標

- 「これなら自分でも出来そう！」と思ってもらうこと



コントリビュートすると良いこと

- 勉強になる

普段CDKを使う際に構成を理解していると少しは早く実装できる
CloudFormationの制約かCDKの問題かの区別が付きやすい

- 何の機能ができるのかいち早く知れる

先日発表されたCLFからCDKのマイグレーションやv2の詳細を先に知れる

- 自分のソースが意外なところで出たりして面白い



CDK Dayの様子

- Building Real-time Backends with AWS AppSync & CDK

```
lib > TS cdk-backend-stack.ts > CdkBackendStack > constructor > [e] api
6   export class CdkBackendStack extends cdk.Stack {
7     constructor(scope: cdk.Construct, id: string, props?: cdk.StackProps) {
8       super(scope, id, props);
9
10      const api = new appsync.GraphqlApi(this, 'Api', {
11        name: 'cdk-notes-appsync-api',
12        schema: appsync.Schema.fromAsset('graphql/schema.graphql'),
13        authorizationConfig: {
14          defaultAuthorization: {
15            authorizationType: appsync.AuthorizationType.API_KEY,
16            apiKeyConfig: {
17              expires: cdk.Expiration.after(cdk.Duration.days(365))
18            }
19          },
20          // additionalAuthorizationModes: {}
21        },
22        xrayEnabled: true,
23      });
24    }
25  }
```

<https://www.youtube.com/watch?v=qJutZqXMdgM&t=4616s>



AWS CDKの構成と詳細

AWS CDK(Cloud Developer Kit)とは

- 使い慣れた言語でAWSリソースをプロビジョニングできるツールキット
- 現在はTS/JS、Python、Java、C#で使用可能(今後はGoも?)
ツールの内部の開発はTypeScriptメイン
- プログラミング言語でリソース管理ができる
→真のIaCツール(個人の感想です)
- いずれは CDK for TerraformでGCPやAzureも管理?



AWS Cloud
Development Kit

AWS CDKの個人的に思う利点(BlackBeltより一部引用)

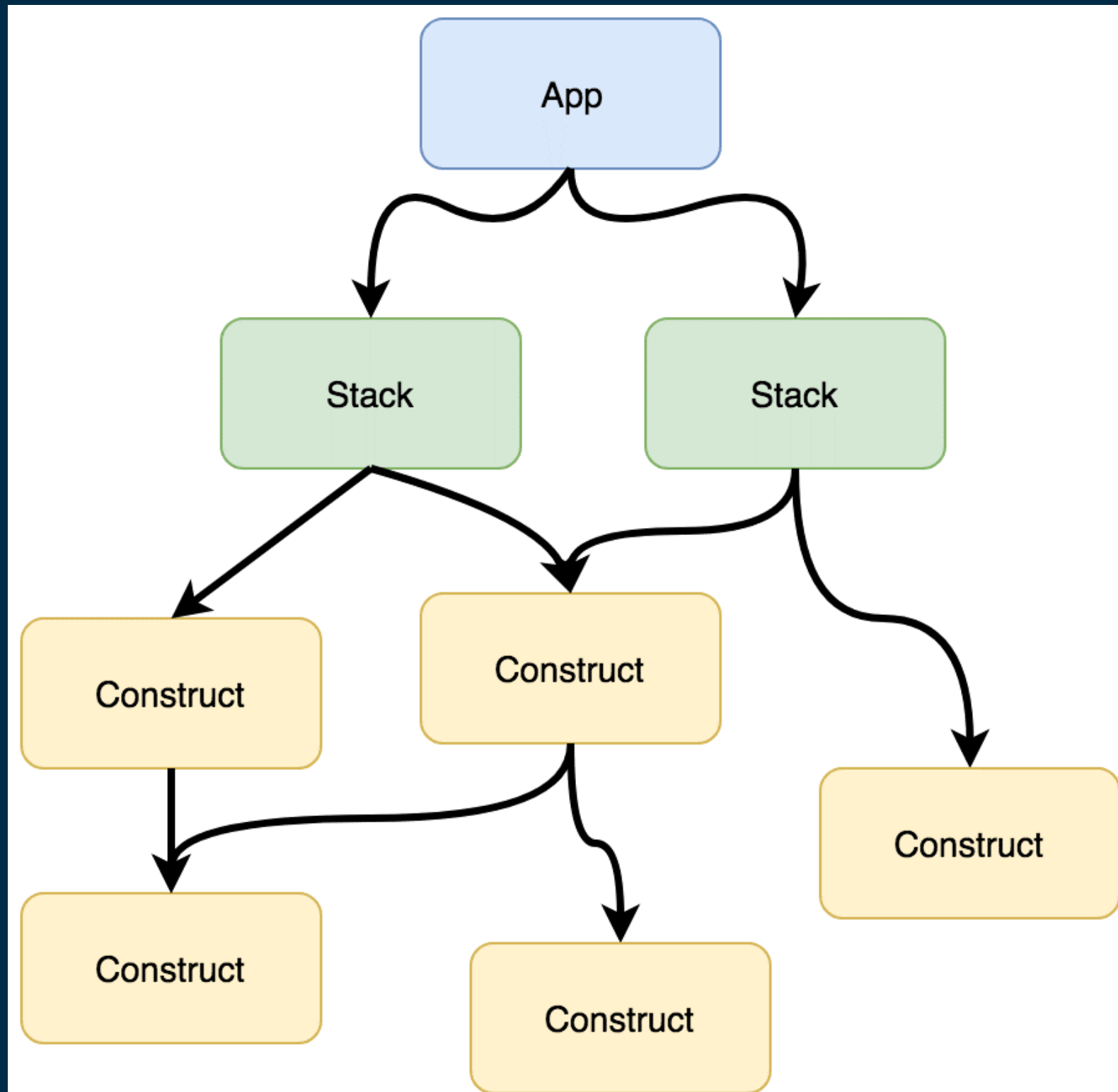
- エディタで型チェック、サジェスト、仕様参照ができる
→Webドキュメントなしでも仕様の確認や実装が可能
- 一般にコード量が減る
→SQS+Lambda+SNSがCloudFormationの実装だと340行程
CDKだと100行程※
- 複数スタック間の依存関係が記述できる
→簡易なコードでAPI Gateway+Lambdaなどを別スタックに書けるなど

※ 「Moving from CFN YAML to CDK - Benefits, how to do it in prod and more」
<https://www.youtube.com/watch?v=qJutZqXMdgM&t=12485s>



AWS CDK アプリケーションの構成

- App
 - CloudFormationテンプレート生成とデプロイに使用される要素
 - スタック間の依存関係を定義
- Stack
 - CloudFormationでいうスタック
 - デプロイの最小単位
- Construct
 - Stack内で生成するAWSリソース
 - Construct Libraryで定義

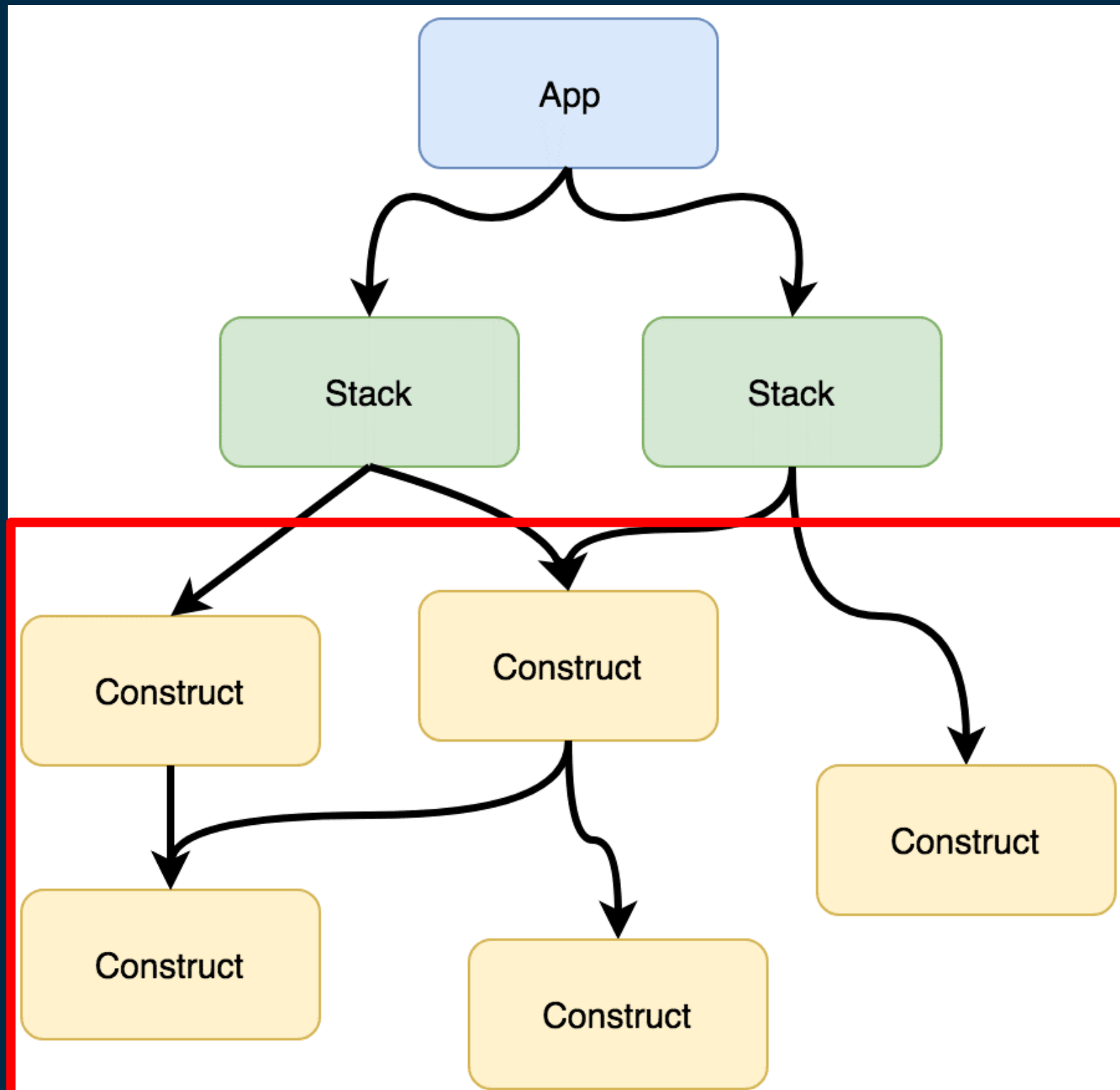


引用元:

<https://aws.amazon.com/jp/blogs/aws/boost-your-infrastructure-with-cdk/>

AWS CDK アプリケーションの構成

- App
 - CloudFormationテンプレート生成とデプロイに使用される要素
 - スタック間の依存関係を定義
- Stack
 - CloudFormationでいうスタック
 - デプロイの最小単位
- Construct
 - Stack内で生成するAWSリソース
 - Construct Libraryで定義

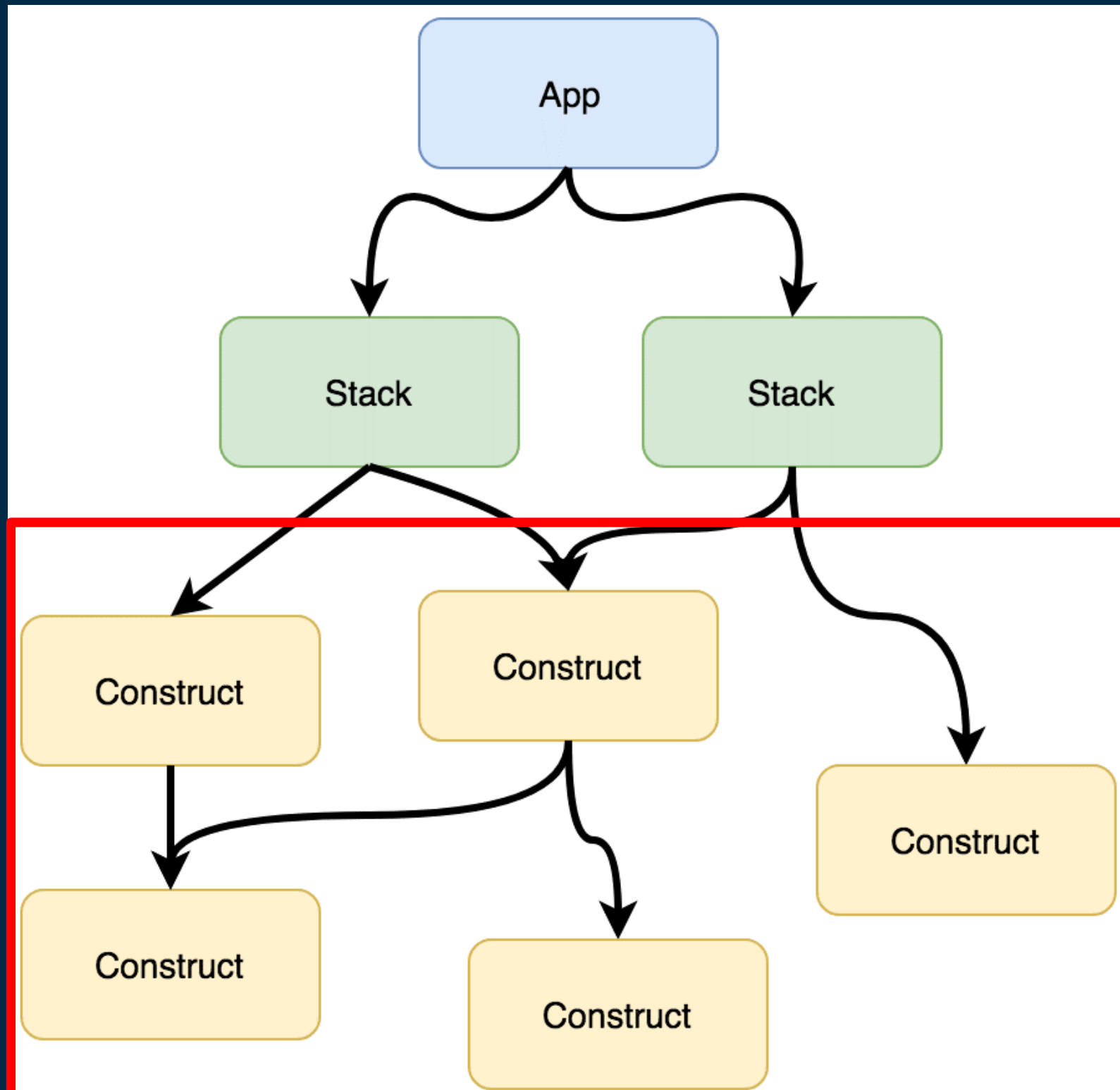


引用元:

<https://aws.amazon.com/jp/blogs/aws/boost-your-infrastructure-with-cdk/>

Constructの構成

- L1 (Low Level) Construct
CloudFormationのリソースと
1対1で対応 (自動生成)
- L2 (High Level) Construct
デフォルト値や追加の関数を実装し
てL1 Constructを抽象化
- L3 Construct (Patterns or etc...)
複数リソースやL2 Constructをさら
に抽象化して利用する場合



TIPS: Constructの構成(AppSyncの具体例)

- ベースとなるソース

Githubリポジトリ : <https://github.com/aws/aws-cdk>

上記リポジトリの `aws-cdk/packages/@aws-cdk/aws-appsync`

`@aws-cdk`配下に各AWSリソース用のパッケージがある

- L1 (Low-Level) Construct

`aws-appsync/lib` 配下の `appsync.generated.ts` ファイル

ビルドしないと生成されない(GitHub上で見えない)ので Gitpod などで確認

- L2 (High-Level) Construct

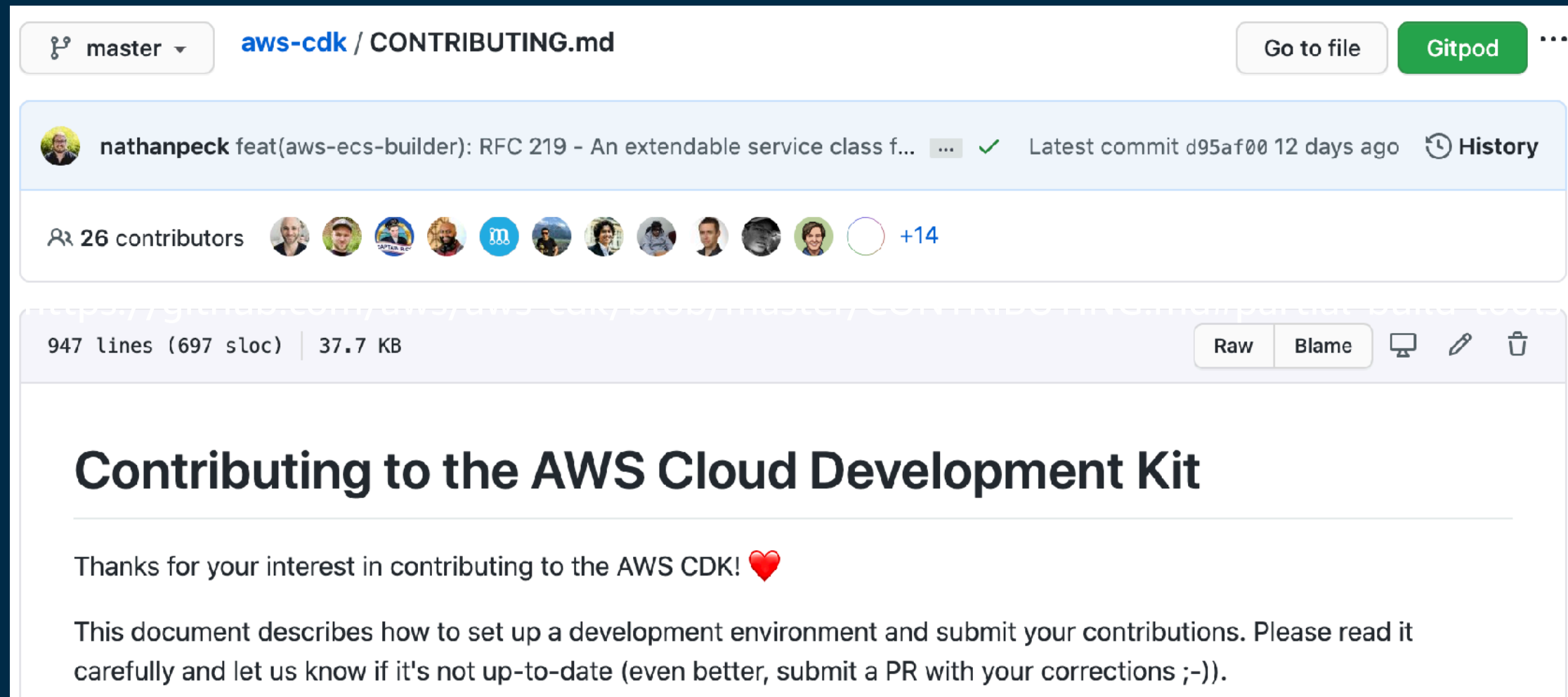
`graphqlapi.ts`、`schema-base.ts` などいくつかのファイルに分かれている



コントリビューティングガイドの説明

コントリビューティングガイドとは

- OSSへ貢献したい人向けに設計や実装方式を共有するドキュメント



The screenshot shows the GitHub interface for the file `CONTRIBUTING.md` in the `aws-cdk` repository. The file is in the `master` branch. The commit history shows a commit by `nathanpeck` titled "feat(aws-ecs-builder): RFC 219 - An extendable service class f..." with a checkmark, committed 12 days ago. Below the commit information, there are 26 contributors listed with their profile pictures. The file statistics show 947 lines (697 sloc) and 37.7 KB. The file content is displayed below, starting with the title "Contributing to the AWS Cloud Development Kit".

master aws-cdk / CONTRIBUTING.md Go to file Gitpod

nathanpeck feat(aws-ecs-builder): RFC 219 - An extendable service class f... ✓ Latest commit d95af00 12 days ago History

26 contributors +14

947 lines (697 sloc) | 37.7 KB Raw Blame

Contributing to the AWS Cloud Development Kit

Thanks for your interest in contributing to the AWS CDK! ❤️

This document describes how to set up a development environment and submit your contributions. Please read it carefully and let us know if it's not up-to-date (even better, submit a PR with your corrections ;-)).

<https://github.com/aws/aws-cdk/blob/master/CONTRIBUTING.md>



コントリビューティングガイドの内容

- PRを送るまで段階ごとにやるべきことなどが書かれている

今回は以下の3つを中心に紹介

- 環境構築
- issueから実装マージまでの流れ
- 各種ツールの紹介



環境構築

Gitpod

GitHubからワンクリックで環境構築して使えるWebIDE(今回紹介)
→ソースの確認やちょっとした修正で有用

Docker

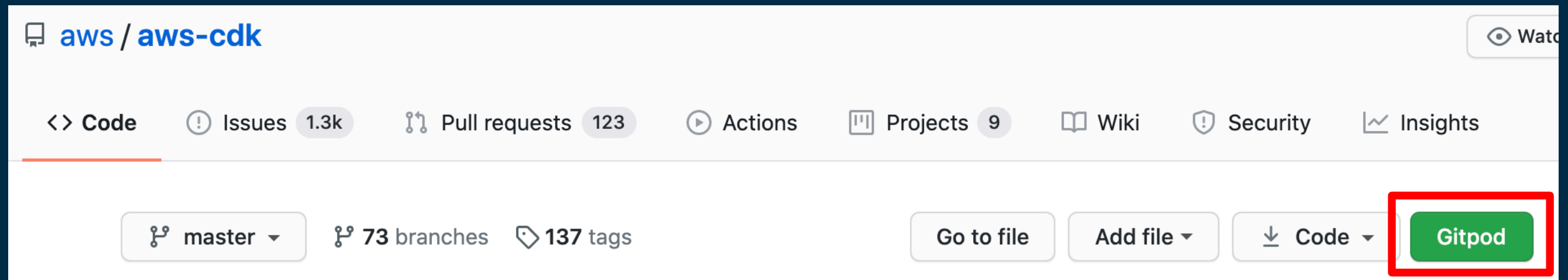
Dockerイメージから環境作成
→継続的な開発で有用

ローカル環境

Node.jsやJavaのバージョンを合わせて構築
→環境の管理が大変なので多分非推奨

Gitpodとは

- GitHubからワンクリックで環境構築して使えるWebIDE
 - VS Code ライクな開発環境がWeb上で使える
 - 初期のビルド設定もあるのでワンクリックで実装が始められる
 - 1ヶ月で50時間までは無料



TIPS: GitpodをCDKの開発で使う場合の注意点

- Gitpod初回起動は1時間弱かかる

全体ビルドが走るので立ち上がるまでが遅い。1回ビルドが通れば再起動時は数分程度で動作可能

2回目以降の起動はGitpodの画面で行う。(先ほどのボタンを押すと毎回新しいインスタンスができる)

- 途中でビルドがこけると別のリソースのせいでテストが通らない時もある

対象のディレクトリにて `scripts/buildup`(後述) で関連するリソースを再度ビルドすると多分大丈夫

最悪新しいGitpodの環境を立ち上げる

- Gitpod上のAccess Control※で権限を与えないとPushできない

GitpodのWebUIで「write public repos」の権限を与えること

※<https://gitpod.io/access-control/>



issueから実装マージまでの流れ

- コントリビューティングガイドをベースに少し変えています
 - issueを立てる/探す
 - 設計(オプション)
 - 実装(Magic)
 - ▶ 実装、単体/結合テスト
 - ▶ 動作確認
 - コミット作成ルールを確認
 - PRの作成
 - レビュー/マージ

コントリビューティングガイドの流れ

- Pull Requests
 - Pull Request Checklist
 - Step 1: Open Issue
 - Step 2: Design (optional)
 - Step 3: Work your Magic
 - Step 4: Commit
 - Step 5: Pull Request
 - Step 6: Merge

issueを立てる/探す

- 既存のissueを確認して、新しい問題であればissueを立てる

着手できそうであればissueに書く

- 既存のissueで貢献できそうな内容を探す

effort/small、good first issue ラベルが付いているissueは比較的簡単

AWSリソースの改修の場合は「@aws-cdk/aws-xxx」のタグを探す

effort/small	Small work item – less than a day of effort
feature-request	A feature should be added or improved.
good first issue	Related to contributions. See CONTRIBUTING.md

label 検索結果

Dzhuneyt commented on 28 Apr

Now that X-Ray exists as a feature that can be enabled on [aws/whats-new/2020/02/aws-appsync-releases-integratio](#) sense to allow it to be enabled through the CDK construct

In other words, something like this should be possible:

```
new GraphQLApi(this, `graphql`, {
    name: `graphql`,
    tracing: Tracing.ACTIVE,
});
```

🙌 I may be able to implement this feature request

⚠️ This feature might incur a breaking change

This is a 🚀 Feature Request

Feature Request の例

設計

- 中~大規模な変更の場合はあらかじめ設計を行う

設計ガイドラインを確認の上でissueにコメントする

→設計思想がなくてない修正だらけに... 最初はわかりやすいissueから着手の方が良いかも

- 設計ガイドラインの確認は有用

CDK全体の設計思想について書かれています (少し長い)

https://github.com/aws/aws-cdk/blob/master/DESIGN_GUIDELINES.md



実装(Magic)

- 一般的なコーディングスタイルで実装

2 space indent、横幅120文字など詳細はガイド参照

- すべての変更には単体テストが必要

test配下のソースに単体/結合テストを追加する(Jestで実装)

- コメント追加も場合によっては必要（修正しないとLintでwarningになる）

外部から呼び出す際のパラメータ変更などAPIの変更はREADMEを更新

パラメータ追加などの場合はjsdocsも更新



実装(単体テスト)

- テスト実装にはJestを使用(一部古いものはnode-unit)
 - JavaScript/TypeScriptで使えるテストフレームワーク
 - 他のフレームワーク(JUnitなど)を使ったことがあれば分かりやすい
- 単体テストの構成
 - 以下のテストが大半
 - Stackを生成して、StackにConstructを作成
 - Stackに想定したResourceが含まれているかチェック



実装(単体テスト: appsync.test.tsの場合)

- Stackを生成して、StackにConstructを作成

```
let stack: cdk.Stack;
let api: appsync.GraphqlApi;
beforeEach(() => {
  stack = new cdk.Stack(); // 普段のbin配下の実装と同様
  api = new appsync.GraphqlApi(stack, 'api', { // lib配下と同様
    authorizationConfig: {},
    name: 'api',
    schema:
      appsync.Schema.fromAsset(path.join(__dirname,
        'appsync.test.graphql')),
  });
});
```

実装(単体テスト: appsync.test.tsの場合)

- テストしたいConstructを作成
- Stackに想定したResourceが含まれているかチェック

```
test('appsync should configure pipeline ... contents', () => {  
  // WHEN  
  new appsync.Resolver(stack, 'resolver', {  
    api: api,  
    typeName: 'test',  
    fieldName: 'test2',  
    pipelineConfig: ['test', 'test'],  
  });  
  // THEN  
  expect(stack).toHaveResourceLike('AWS::AppSync::Resolver', {  
    Kind: 'PIPELINE',  
    PipelineConfig: { Functions: ['test', 'test'] },  
  });  
});
```


実装(結合テスト)

- 複数サービスに跨る変更やCLFを使った新しい機能を作る時に必要
- 結合テストの種類
 - CloudFormationテンプレートを生成して比較
cdk synthを実行してtest配下の *.expected.json ファイルと比較
 - デプロイして検証
yarn integ コマンドを実行するとAWSアカウントにデプロイ (AWSの認証情報設定は必要)



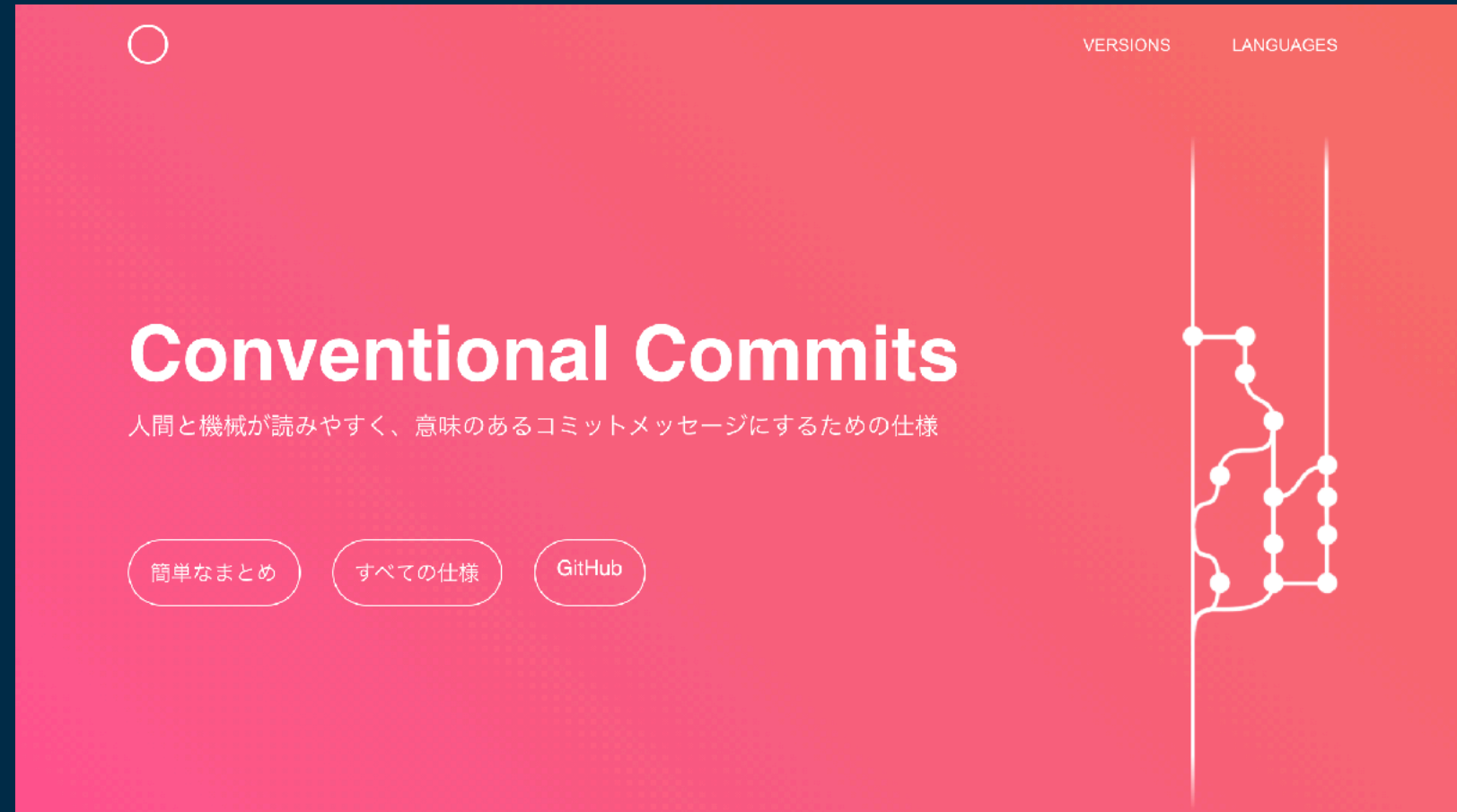
コミット

- タイトルは以下の文言で始める

```
feat(module): title  
fix(module): title  
refactor(module): title  
chore(module): title
```

- 変更が自明でない場合は
確認内容を記載

- 重大な変更がある場合は記載



コミットメッセージ作成のベースは「Conventional Commits」

<https://www.conventionalcommits.org/ja/v1.0.0-beta.4/>

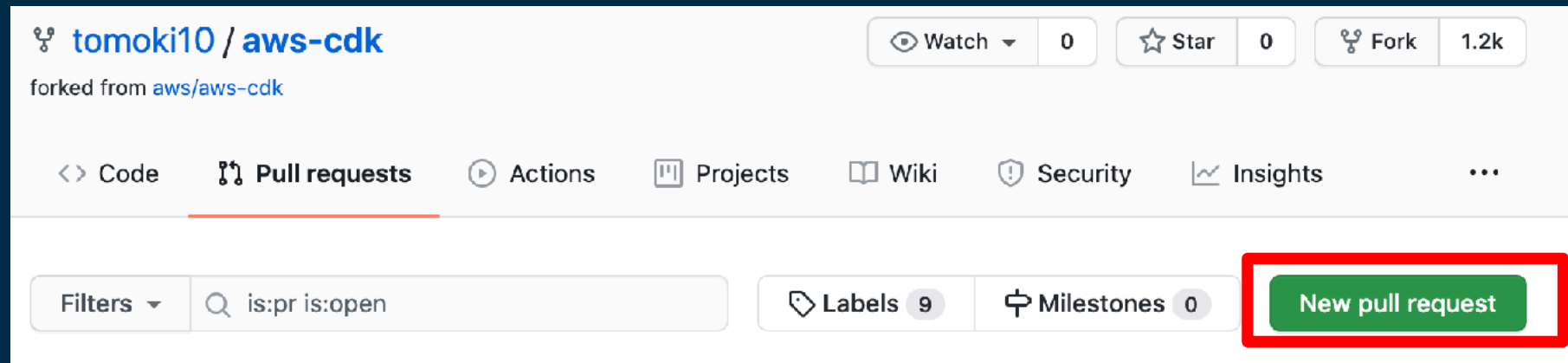
Pull Request作成

- GitHubの画面上などからでPull Requestを作成

普段GitHubフローなどで開発している場合はそんなに変わらない

- PRチェックリストに従ってセルフチェック

- 単体テストは追加したか
- jsdocsは書いたか
- READMEは適宜更新したか etc...



レビュー/マージ

- メンテナからレビューを受ける

PRに対して指摘があれば適宜修正する

早いとPR作成した当日にレビュー内容がコメントされる

- ソースを修正してPush

修正に問題がなければメンテナがapproveする

※他の修正が並行で走っていてテストがコケる場合もあるのでその際は修正が必要

→完了🎉

TIPS: 内部のツール (一部抜粋)

- Linter
 - ESLint
 - pkglint
 - awslint
 - prlint
- scripts/buildup
- cfn2ts



TIPS: 内部のツール (Linter)

- ESLint

tools内の eslintrc.js の設定が全体に反映。GitpodでもESLint Pluginの追加で警告を出せる

- pkglint

tools内の rules.ts の設定に応じて、ドキュメント形式 や License の整合性を検証

- awslint

aws construct library用のlinter。設計ガイドラインに準拠しているかを検証

(ExperimentalなパッケージだとJSDocがちゃんと付いてなくてwarnが出てることも)

- prlint(ベータ)

PRの命名規則やコメントに応じて必要な項目が足りているのか検証



TIPS: 内部のツール (script/buildup)

- scripts/buildup

パッケージのディレクトリ内で実行すると、関連するパッケージだけ再ビルドできる

初回ビルドは大体1時間くらいかかってコケることがあるので使える

```
lerna success - cdk-build-tools
lerna success - cdk-integ-tools
lerna success - cfn2ts
lerna success - nodeunit-shim
lerna success - pkglint
lerna success - pkgtools
lerna success - prlint
lerna success - yarn-cling
```

```
real    46m36.159s
user    93m8.947s
sys     5m19.583s
Done in 2843.95s.
```

実際に初回起動した際の結果

TIPS: 内部のツール (cfn2ts)

- CloudFormationリソースからTSのコードを生成できるツール

***.generated.tsファイルはこのツールで生成されている。

L1 Constructの対応だけやけに早いのはこのツールのおかげ (Chatbotなど)

- ビルド時に jsonファイル から TSファイルを生成

@aws-cdk/cfnspec 配下にビルドの際生成される specification.json を見ている？



実際コントリビュートした際の流れ

実際コントリビュートした際の流れ

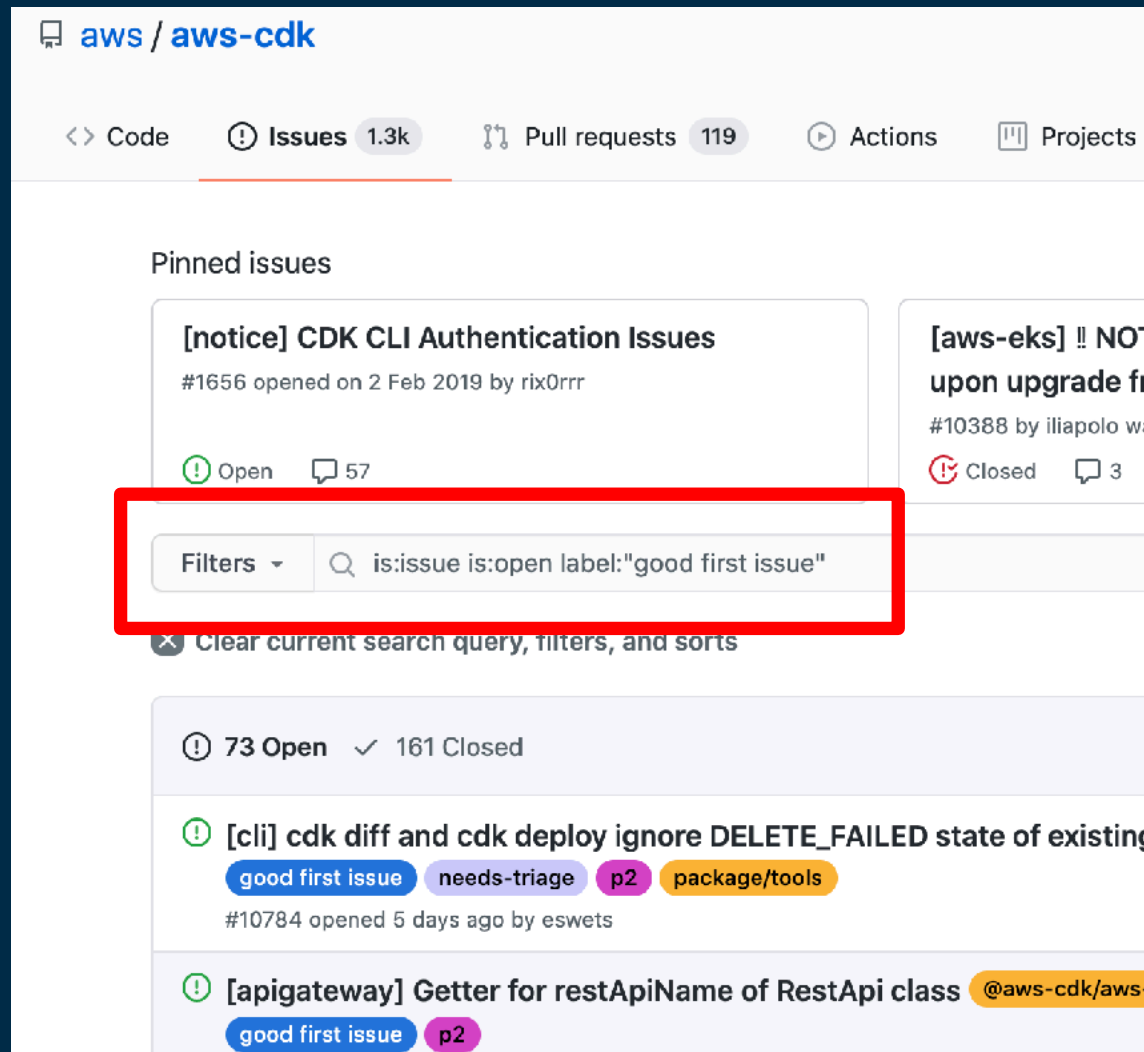
- issueを探す/コメントする
- Fork/Gitpod設定
- 実装
- 単体/結合テスト作成
- Branch作成/Push
- Pull Request作成
- レビュー/修正
- メンテナがマージ

※あくまで参考

この通りでなくとも大丈夫です

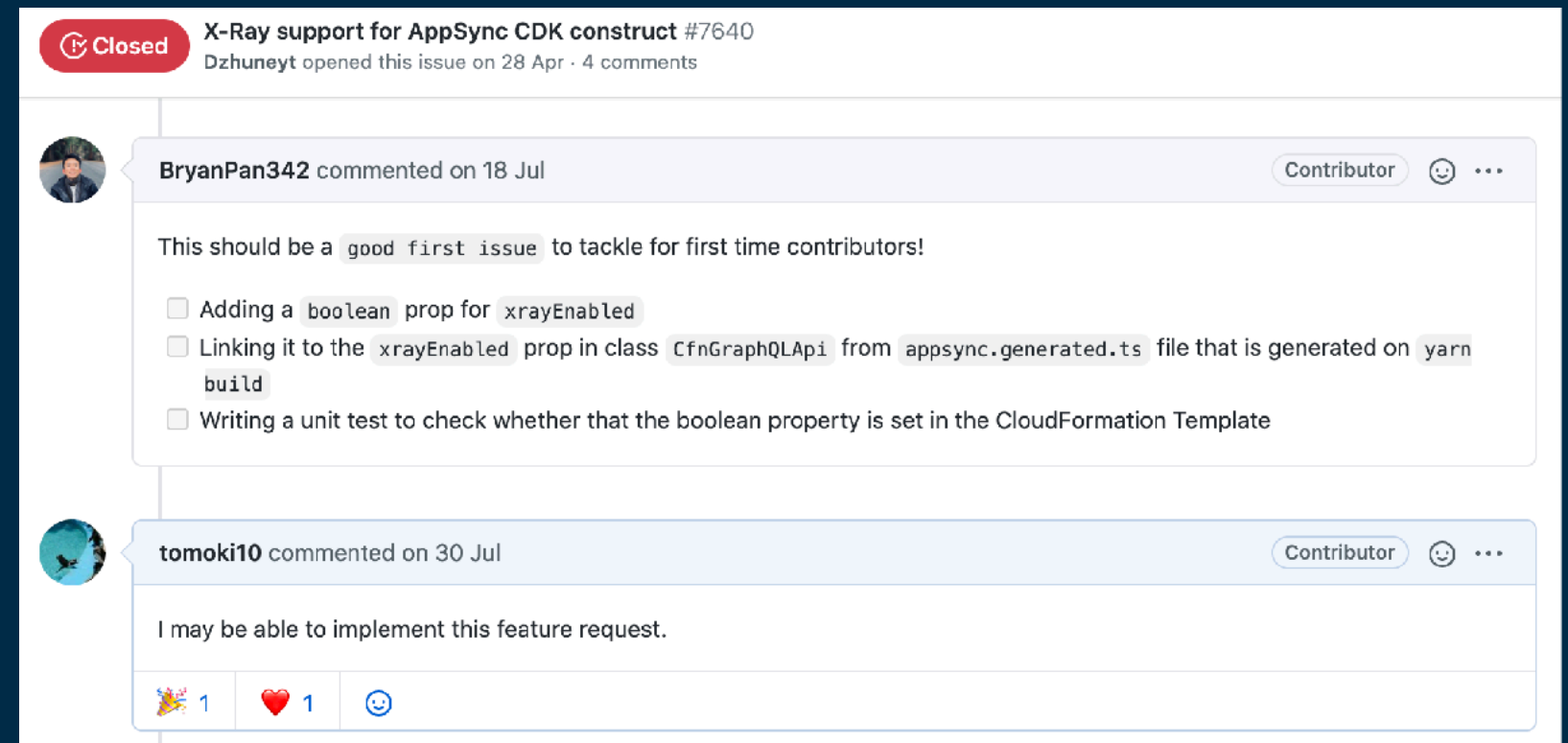
issueを探す/コメントする

- issueを探す



The screenshot shows the GitHub repository page for `aws/aws-cdk`. The navigation bar includes links for Code, Issues (1.3k), Pull requests (119), Actions, and Projects. Under the 'Issues' section, there are 'Pinned issues' and a search bar. The search bar is highlighted with a red box and contains the filter query: `is:issue is:open label:"good first issue"`. Below the search bar, there are statistics for 73 Open and 161 Closed issues. The first issue listed is `[cli] cdk diff and cdk deploy ignore DELETE_FAILED state of existing`, which is labeled as a 'good first issue' and has other labels like 'needs-triage', 'p2', and 'package/tools'.

- issueにコメントする



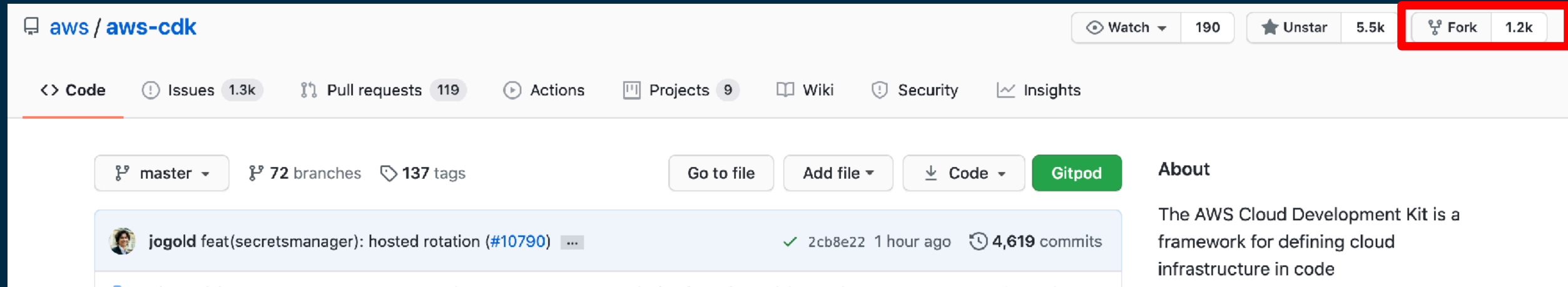
The screenshot shows a GitHub issue page for `X-Ray support for AppSync CDK construct #7640`, which is marked as 'Closed'. The issue was opened by Dzhuneyt on 28 Apr and has 4 comments. The first comment is from BryanPan342, dated 18 Jul, and says: "This should be a `good first issue` to tackle for first time contributors!". Below the comment is a checklist of tasks:

- Adding a `boolean` prop for `xrayEnabled`
- Linking it to the `xrayEnabled` prop in class `CfnGraphQLApi` from `appsync.generated.ts` file that is generated on `yarn build`
- Writing a unit test to check whether that the boolean property is set in the CloudFormation Template

The second comment is from tomoki10, dated 30 Jul, and says: "I may be able to implement this feature request." Below the comment are reaction icons for 'celebrate' (1), 'heart' (1), and 'smiley face'.

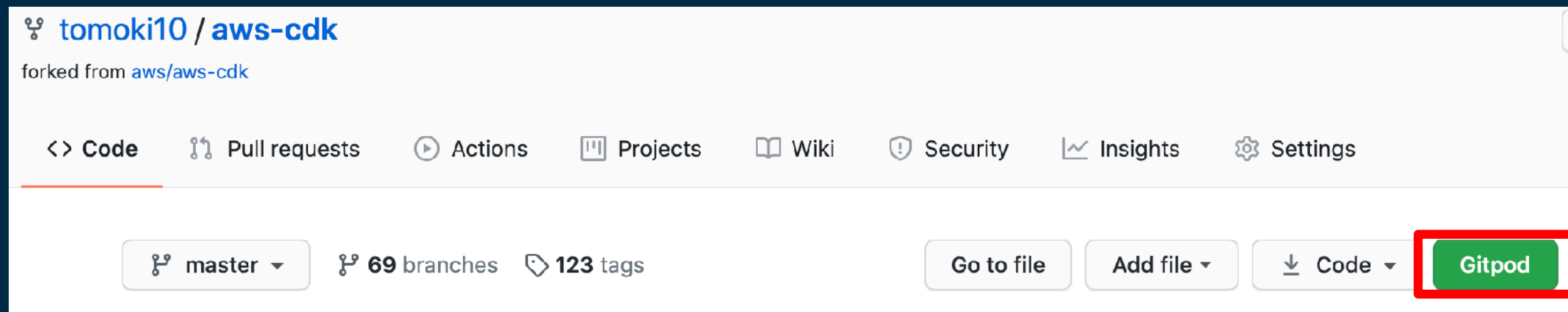
Fork/Gitpod設定

- リポジトリからForkする



The screenshot shows the GitHub repository page for `aws/aws-cdk`. The repository has 190 watchers, 5.5k stars, and 1.2k forks. The `Fork` button is highlighted with a red box. The repository has 72 branches and 137 tags. A recent commit by `jogold` is visible, along with a `Gitpod` button.

- ForkしたリポジトリでGitpodを起動



The screenshot shows the GitHub repository page for `tomoki10/aws-cdk`, which is a fork of the original repository. The repository has 69 branches and 123 tags. The `Gitpod` button is highlighted with a red box.



実装

- Gitpod上でソースを確認して実装

The screenshot shows the Gitpod IDE interface. On the left, the Explorer view displays a file tree for an AWS-CDK project, including folders like 'aws-appstream', 'aws-appsync', 'coverage', and 'lib'. The 'lib' folder is expanded, showing files such as 'appsync.generated.d.ts', 'appsync.generated.js', 'appsync.generated.ts', 'data-source.d.ts', 'data-source.js', 'data-source.ts', 'graphqlapi-base.d.ts', 'graphqlapi-base.js', 'graphqlapi-base.ts', 'graphqlapi.d.ts', 'graphqlapi.js', 'graphqlapi.ts', 'index.d.ts', 'index.js', 'index.ts', and 'kev.d.ts'. The 'graphqlapi.ts' file is selected and open in the editor. The editor shows TypeScript code with comments and type annotations. The terminal window at the bottom displays the file system structure of the workspace, showing directories like 'coverage', 'lib', 'node_modules', and 'test', and files like 'jest.config.js', 'package.json', 'tsconfig.json', and 'tsconfig.tsbuildeinfo'. The terminal prompt is 'gitpod /workspace/aws-cdk/packages/@aws-cdk/aws-appsync \$'.



単体/結合テスト

- package.jsonを確認してテスト（yarn build+test の実行結果）

動作がわからない時は、意図的にテストを壊して影響範囲を確認

- 個人用のAWS環境で動作確認

```
>_ gitpod /workspace/aws-cdk/packages/@aws-cdk/aws-appsync ×
===== Coverage summary =====
Statements   : 85.8% ( 435/507 )
Branches     : 91.14% ( 391/429 )
Functions    : 77.36% ( 164/212 )
Lines        : 86.38% ( 406/470 )
=====

Test Suites: 16 passed, 16 total
Tests:       172 passed, 172 total
Snapshots:   0 total
Time:        18.965 s
Ran all test suites.
Verifying integ.api-import.js against integ.api-import.expected.json ... OK.
Verifying integ.auth-apikey.js against integ.auth-apikey.expected.json ... OK.
Verifying integ.graphql-iam.js against integ.graphql-iam.expected.json ... OK.
Verifying integ.graphql-schema.js against integ.graphql-schema.expected.json ... OK.
Verifying integ.graphql.js against integ.graphql.expected.json ... OK.
Tests successful. Total time (24.2s) | /workspace/aws-cdk/node_modules/jest/bin/jest
.js (20.2s) | cdk-integ-assert (4.0s)
Done in 49.15s.
gitpod /workspace/aws-cdk/packages/@aws-cdk/aws-appsync $
```



Branch作成/Push

- Gitpod上でBranch作成してcommitしてPush

Gitを使用した普段の開発と特に変わらないです（Gitpodの権限だけは注意！）

```
Verifying integ.api-import.js against integ.api-import.expected.json ... OK.
Verifying integ.auth-apikey.js against integ.auth-apikey.expected.json ... OK.
Verifying integ.graphql-iam.js against integ.graphql-iam.expected.json ... OK.
Verifying integ.graphql-schema.js against integ.graphql-schema.expected.json ... OK.
Verifying integ.graphql.js against integ.graphql.expected.json ... OK.
Tests successful. Total time (24.2s) | /workspace/aws-cdk/node_modules/jest/bin/jest
.js (20.2s) | cdk-integ-assert (4.0s)
Done in 49.15s.
gitpod /workspace/aws-cdk/packages/@aws-cdk/aws-appsync $ git checkout -b tomoki10/a
ppsync-xray-add
```


Pull Request作成

- GitHub上でPull Requestを作成

The screenshot shows a GitHub Pull Request interface. At the top, the title is "feat(appsync): add x-ray parameter to AppSync #9389". Below the title, there is a green "Open" button and a status bar indicating "10 commits into aws:master from tomoki10:tomoki10/appsync-xray-add on 7 Aug". A navigation bar shows "Conversation 12", "Commits 10", "Checks 5", and "Files changed 3". A comment from user "tomoki10" is displayed, dated "1 Aug". The comment includes a link to "Close: #7640" and a checklist of three items: "Adding a boolean prop for xrayEnabled", "Linking it to the xrayEnabled prop in class CfnGraphQLApi from appsync.generated.ts file that is generated on yarn build", and "Writing a unit test to check whether that the boolean property is set in the CloudFormation Template". At the bottom of the comment, there is a license confirmation statement: "By submitting this pull request, I confirm that my contribution is made under the terms of the Apache-2.0 license".

feat(appsync): add x-ray parameter to AppSync #9389

Open 10 commits into aws:master from tomoki10:tomoki10/appsync-xray-add on 7 Aug

Conversation 12 Commits 10 Checks 5 Files changed 3

tomoki10 commented on 1 Aug Contributor

Close: #7640

- ✓ Adding a boolean prop for xrayEnabled
- ✓ Linking it to the xrayEnabled prop in class CfnGraphQLApi from appsync.generated.ts file that is generated on yarn build
- ✓ Writing a unit test to check whether that the boolean property is set in the CloudFormation Template

By submitting this pull request, I confirm that my contribution is made under the terms of the Apache-2.0 license

mergify bot のlintで弾かれる

- 以下の原因でリジェクト

1コミットでPR作成、機能追加なのにREADME変更なし

mergify bot commented 12 minutes ago

Title does not follow the guidelines of [Conventional Commits](#). Please adjust title

Add more commits by pushing to the `tomoki10/appsync-xray-add` branch on `tomoki10`

- Review required**
At least 1 approving review is required by reviewers with write access. [Learn more](#)
- Some checks were not successful**
2 failing, 2 successful, and 1 pending checks
- PR Linter / mandatory-changes (pull_request)** Failing after 9s — mandatory
- Auto-approve Dependabot / build (pull_request)** Successful in 2s
- Semantic Pull Request** — PR has only one commit and it's not semantic;
- AWS CodeBuild us-east-1 (AutoBuildProject6AEA49D1-qxepHUsryhcu)**
- Summary** — 2 rules match and 2 potential rules

PR Linter / mandatory-changes
failed 12 minutes ago in 9s

Search logs

- ▶ **Set up job** 2s
- ▶ **Checkout** 3s
- ▶ **Install packages** 3s
- ▼ **Validate** 1s
 - 1 ▶ Run `./.github/actions/prlinter`
 - 6 Creating authenticated GitHub Client
 - 7 ⌚ Fetching PR number 9389
 - 8 ⌚ Fetching files for PR number 9389
 - 9 ⌚ Validating...
 - 10 **##[error]Features must contain a change to a README file**
- ▶ **Post Checkout** 0s
- ▶ **Complete job** 0s

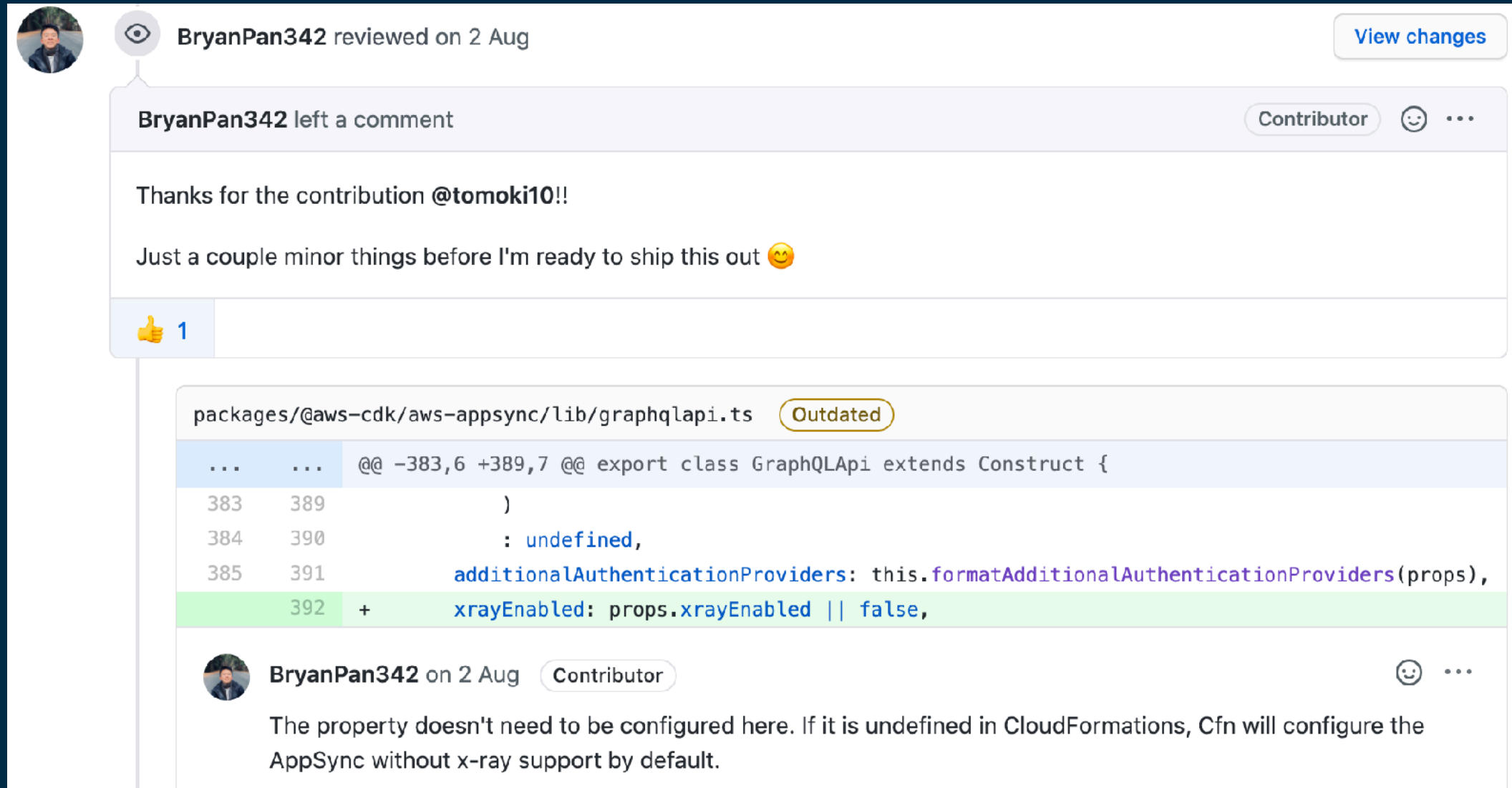
merging bot のlintなどで弾かれる

- エラーの原因を修正して再度Pushすれば完了

The screenshot shows a GitHub pull request for the branch 'add to readme' with commit hash 'aaf8a7b'. The interface includes instructions to push to the 'tomoki10/appsync-xray-add' branch on the 'tomoki10/aws-cdk' repository. A list of checks is displayed, including a 'Review required' error, 'Some checks haven't completed yet' (4 successful, 1 expected), 'Auto-approve Dependabot / build (pull_request)' (Successful in 3s), 'PR Linter / mandatory-changes (pull_request)' (Successful in 37s), 'AWS CodeBuild us-east-1 (AutoBuildProject6AEA49D1-qxepHUsryhcu)' (Expected — Waiti... Required), 'Semantic Pull Request — ready to be squashed', and 'Summary — 1 rule matches and 3 potential rules'. At the bottom, a warning states 'This branch is out-of-date with the base branch' and provides an 'Update branch' button.

レビュー/修正

- メンテナの方から指摘が来るので修正



BryanPan342 reviewed on 2 Aug [View changes](#)


BryanPan342 left a comment [Contributor](#) 😊 ⋮

Thanks for the contribution @tomoki10!!

Just a couple minor things before I'm ready to ship this out 😊

👍 1

```
packages/@aws-cdk/aws-appsync/lib/graphqlapi.ts Outdated  
...    ...    @@ -383,6 +389,7 @@ export class GraphQLApi extends Construct {  
383    389        )  
384    390        : undefined,  
385    391        additionalAuthenticationProviders: this.formatAdditionalAuthenticationProviders(props),  
392    +    xrayEnabled: props.xrayEnabled || false,
```

 BryanPan342 on 2 Aug [Contributor](#) 😊 ⋮

The property doesn't need to be configured here. If it is undefined in CloudFormations, Cfn will configure the AppSync without x-ray support by default.

メンテナがマージ


- メンテナの方1人以上からapproveされればマージ🎉

BryanPan342 left a comment Contributor 😊 ...


LGTM 🎉

Thanks for the contribution @tomoki10!!


...

 ✓ BryanPan342 approved these changes on 7 Aug View changes

...

 mergify bot commented on 7 Aug Contributor 😊 ...

Thank you for contributing! Your pull request will be updated from master and then merged automatically (do not update manually, and be sure to [allow changes to be pushed to your fork](#)).

 mergify bot merged commit 51921ad into aws:master on 7 Aug View details Revert
6 of 7 checks passed

まとめ

まとめ

- AWS CDKの構成と詳細を紹介

CDKはAWSに留まらず発展する可能性がある！ 乗るしかない、このビッグウェーブに！

- コントリビューティングガイドの説明

やり方はコントリビューティングガイドに全て書いてある。読もう！

- 実際コントリビュートした際の流れの紹介

失敗した内容も含めてコントリビュートまでの全体像を紹介

→明日からあなたもAWS CDKのコントリビューターになるう！！！！



Thank you!

Tomoki Sato
sato.tomoki@classmethod.jp