はじめに

本書を手にとっていただきありがとうございます。

Amplify Consoleを利用してWebサイト・Webアプリケーションのホスティングを検討している方や、既に活用されている方に対して、Amplify Consoleを現場で運用して得られた知見を共有したいと考えて本書を執筆しました。

Amplify Consoleを使えば、フロントエンド開発のベストプラクティスを手軽に導入できます。

Amplify Consoleの登場以前は、AWSであればS3、CloudFront、CodeBuildなどを組み合わせたCI/CDパイプラインを自前で構築する必要がありました。

Amplify Consoleの登場によって、それらの構築は不要になりました。それだけでなく、Pull Requestの作成に応じたWeb Previewやコミットごとのキャッシュの自動的な即時無効化などの機能も提供されています。

しかし、Amplify Consoleは登場してから日の浅いサービスでもあります。便利な一方で、通知やビルドの設定で悩まされることは誰もが通る道です。

また、Amplifyファミリー、特にAmplify CLIと混同することで、検索がしづらくなっている面も否めません。

そうした状況で、本書が少しでもAmplify Consoleを実践するための助けになれば幸いです。

筆者は2019年に業務でAmplify Consoleを導入して以来、半年のうちに3つ以上のAWSアカウントで合計20以上のWebアプリケーションをセットアップしてきました。

少人数のチームで多くのアプリケーションを管理するために、Amplify Consoleの機能の活用、CloudFormationおよびAWS CDKを使った一括管理、様々なトラブルシューティングを行ってきました。

本書では、そうした経験から得られたAmplify Consoleを扱う上での知見を共有します。

これからAmplify Consoleに触る方も、既にAmplify Consoleを導入されている方も、Amplify Consoleを実践する上で何らかの発見があれば幸いです。

本書で扱うこと

Amplify Consoleの基本的な機能

Amplify Consoleを実践するための知見

Amplify Consoleに関するトラブルシューティング

本書で扱わないこと

Amplify Consoleのチュートリアル

Amplify Framework（Amplify CLIなど）を利用した開発

本書の構成

基本的な機能を紹介する前半と、実践した上での知見をご紹介する後半に分かれています。

前半の第1章では静的WebホスティングとAmplify Consoleについて、第2章ではAmplify Consoleの基本的な機能についてご紹介します。

後半の第3章ではAmplify ConsoleとCloudFormation・AWS CDKについて、第4章では実践する上でのテクニックをご紹介します。また、第5章では筆者が対応したトラブルシューティングを簡単にご紹介します。

意見と質問

本書はAWSドキュメントやサポートに確認した内容をもとに執筆していますが、誤りや不正確な表現に気づかれることもあるかと思います。また、本書で紹介するやり方よりもよいやり方をご存知の方もいるかもしれません。

そうした点については、今後の版で随時更新させていただきたいのでお知らせいただけると幸いです。ご意見だけでなく質問も大歓迎です。連絡先を以下に記します。

Mail: hiroga1030@gmail.com

Twitter: @hiroga\_cc

入門Amplify Console

Amplify Consoleは、AWSの静的Webホスティングサービスです。

NetlifyやVercel、Firebase Hostingらと同様に、GitHubなどと連携したCI/CDパイプライン・ホスティング・CDNを備えています。

それらのサービスとの対比に加え、AWSのサービスの中でどう位置づけられるかも合わせて見ていきましょう。

Amplify Console

静的Webホスティングサービス

本書における静的Webホスティングサービスとは、Gitリポジトリと連携したCI/CDパイプラインを備え、SPAや静的サイトをデプロイできるサービスをいいます。

代表的な静的WebホスティングサービスであるNetlifyは、2015年4月に、MakerLoop, Inc.（現: Netlify, Inc.）からリリースされました。

Netlifyは日本では2016年頃から徐々に利用されているようです。

例えば、Netlifyに関する一番古いQiitaの記事は2016年8月18日に投稿されています。

当時はAmplify Consoleがなかったので、AWSでCloudFront+S3を運用している方がNetlifyのシンプルさに惹かれるということもあったようです。

高機能ホスティングサービスNetlifyについて調べて使ってみた

HTTPSの静的コンテンツをホストするならs3よりNetlifyが俺の求めていたものだった

ちなみにNetlifyのインフラはAWSです。

Amplify Console

Amplify Consoleは、2018年11月にAWSからリリースされた静的Webホスティングサービスです。2019年4月に東京リージョンでも使えるようになりました。

CloudFrontとS3とCodeBuildの設定を自動化したサービスと言えるかもしれません。

ただし、CloudFrontのディストリビューションやS3のバケットはAWSが管理するAWSアカウント内に作成されるので、私達が気にする必要はありません。

Amplify Console以外の静的Webホスティングサービス

2020年9月現在、大手クラウドベンダーのサービスを含めた複数の静的Webホスティングサービスがあります。

AWS Amplify Console

Firebase Hosting

Azure Static Web Apps

Netlify

Vercel

Document History for AWS Amplify

2020年4月にZEITから改称

AmplifyファミリーとAmplify Console

AWS Amplifyは、AWSがフロントエンド向けに提供するサービスを総称するブランド名です。Amplify Consoleは、そのうち静的サイトホスティングを担うサービスに過ぎません。

AWS Amplifyブランドで提供されるサービスは以下の通りです。

Amplifyフレームワーク - Amplifyライブラリー（SDK）

Amplifyフレームワーク - AmplifyUIコンポーネント

Amplifyフレームワーク - Amplify CLI

Amplify Console

特に注意したいのは、Amplifyによる静的WebホスティングサービスはAmplify ConsoleとAmplify CLIの2つがあることです。

Amplify CLIはいわばブートストラップで、ユーザーの質問に対して動的にCloudFormationのテンプレートを生成・デプロイします。

その中には、CloudFront+S3やAmplify Consoleによる静的Webホスティングも含まれます。

Amplify Consoleについて検索していたら、いつの間にかCloudFront+S3を構築していた...とならないよう、Amplifyに関する記事を読むときはAmplify CLIとAmplify Consoleのどちらを指しているのか注意してください。

静的ウェブホスティングという分類について

本書を書くにあたって悩んだことのひとつが、静的ウェブホスティングという呼び方です。実は、ベンダーごとに呼び方が異なります。

各ベンダーが自社サービスをどのように呼んでいるか見てみましょう（一部筆者訳）

静的ウェブホスティングサービス（AWS）

ウェブ コンテンツ ホスティング（Firebase）

GitHub リポジトリから Azure にフルスタック Web アプリを自動的に構築してデプロイするサービス（Azure）

モダンなWebプロジェクトのAll-in-Oneプラットフォーム（Netlify）

ワークフローと完璧に合致する静的サイトとサーバレスファンクションのクラウドプラットフォーム（Vercel）

ご覧の通り、標準的な呼び名が定まっていません。

悩んだ挙げ句、「この本はAmplify本なんだから、AWSでの呼び名を採用すればいいや」という半ば思考停止的な理由で本書では "静的ウェブホスティング" を用いることにしました。

もし将来標準的な呼び名が決まったら、改訂版を出そうと思います。笑

Amplify Consoleの機能

AWS Amplify Consoleの機能は、インテグレーション・ビルド・ホスティングの3種類に分類できます。

3つの分類ごとに機能を紹介し、特に筆者がよく使うものについては詳細に解説します。

インテグレーション

Amplify Consoleは、マネジメントコンソールから手軽にGitHubなどのリポジトリと接続したワークフローを構築できます。

それらの接続は、Amplify ConsoleのビルドのトリガーとなるWebhookをGitプロバイダーに登録する形で実装されています。

したがって、Webhookを利用して任意のタイミングでビルドを実行することもできます。機能は以下の通りです。

Incoming Webhookによるビルド実行

Gitプロバイダーとの連携機能(GitHub, BitBucket, GitLab, AWS CodeCommit)

Pull RequestごとにWeb Previewをデプロイ

ビルド成果物を保存しているストレージを直接指定(Manual Deploy)したデプロイ

Incoming Webhookのユースケース

まず考えられるのは、リポジトリのソースコード以外の更新をトリガーにWebコンテンツを最新化したいケースではないでしょうか。

例えば、ヘッドレスCMSでコンテンツに更新があった場合や、Git SubmoduleやGit Subtree、npmモジュールが更新された場合などが考えられます。

ヘッドレスCMSサービスによっては、こうしたユースケースが想定されています。例えばContentfulでは、コンソールからリクエストするWebhookを登録することができます。

他には、CircleCIなどの外部サービスでCI/CDを実行したいケースが考えられます。

段階的にAmplify Consoleに移行したい場合などが当てはまるでしょうか。

Pull Requestの Web Previewについて

GitHubなどで作成されたPull Requestごとに、成果物を一時的なURLにデプロイしてWebでPreviewできる機能です。

GitHubでWeb Previewを有効にするには、対象のリポジトリに対してアクセス権のあるGitHubユーザーがAmplify Consoleに対して権限を付与する必要があります。

チームで運用する場合は、GitHub側にシステムユーザーを作成して作業するのがよいでしょう。

便利な一方で、Web Previewには悩みもあります。

例えばビルドに失敗した場合、原因を知るためにAmplify Consoleにアクセスしなければいけません。GitHubのコンソールには原因が表示されないためです。

また、短い間隔でコミットを連投した場合の挙動が不安定です。詳細はビルドに関するトラブルをご覧ください。

ビルド

一般的なCIサービスに求められる要件を満たしていると言えそうです。

Cypressと統合されたEnd-to-EndテストでUIの証跡を残せる点はポイントかもしれません。

ビルド・テスト・デプロイ・キャッシュ利用・環境変数の仕組み

ビルドに用いるコンテナイメージ・IAMロールのカスタマイズ

ビルドの通知

ビルドの通知の悩み

マネジメントコンソールからメール通知を設定できます。

また、EventBridgeでAmplify Consoleのビルドイベントをフックすることで、Lambdaなどの任意のサービスを動かすことができます。メール以外の通知で解説します。

ホスティング

CloudFrontとS3をベースに、Basic認証やキャッシュの即時無効化などの機能が提供されています。

ちなみにBasic認証ではブランチごとにパスワードを変更でき、またパスワードそのものの有無も設定できます。

Webコンテンツのホスティング・CDN

Basic認証によるアクセス制限

カスタムヘッダーの設定

リダイレクトの設定

アクセスログ

Route53の自動設定

なお、Route53の自動設定を行うには、Amplify Console アプリケーションと指定したドメイン名を登録しているHostedZoneが同じAWSアカウントに存在する必要があります。

外部のドメイン管理サービスや別のAWSアカウントのRoute53で管理している場合は、別途手動で設定が必要です。

Amplify Console vs CloudFront+S3の機能比較

Amplify Consoleのパフォーマンスは、CloudFront+S3 と同レベルです。しかし、機能の違いがあります。

Amplify Consoleの優位点は、コミットごとの即時キャッシュ無効化・ブランチごとの自動デプロイ・Pull Request のWeb Preview 作成、パスワード保護などがマネージドサービスとして提供されていることです。

一方で、CloudFrontでは可能だったLambda@Edgeの利用は現在サポートされていません。

したがって、IPアドレスによるアクセス制限・サーバーサイドレンダリングなどを実装することができません。

Amplify Consoleの前段にCloudFrontを配置してLambda@Edgeを利用する、という方法はあります。

Amplify Consoleを支援するツール

マネジメントコンソール以外でAmplify Consoleを構築・運用するためのツールをご紹介します。

具体的には、AWS CLI・Amplify CLI・CloudFormation・AWS CDK・Terraformを取り上げます。

AWS CLI

AWS CLIからAmplify Consoleを操作できます。念のために付け加えると、AWS CLIとAmplify CLIは全く別のツールです。

AWS CLIでAmplify Consoleアプリケーションを作成するケースは少ないと思います。

現場で利用するケースとしては、マネジメントコンソールで作成したAmplify Consoleアプリケーションの参照や、アクセスログの取得などが考えられます。

Amplify CLI

Amplify CLIでもAmplify Consoleのアプリケーションを構築・管理することができます。

本書では詳しく取り上げませんが、Amplify Console以外のAWSサービスをまとめて構築したい場合に便利かもしれません。

CloudFormation & AWS CDK

マネジメントコンソール以外からAmplify Consoleアプリケーションを管理する場合に、以下の理由から最もおすすめできるツールです。

バックエンドとの整合性を考慮すると、dev, prodなど環境ごとに同じ構成のAmplify Consoleアプリケーションを作成するのがベストであるため。

Slack通知のために、Amplify Console Appと同時にEventBridgeのRuleを作成したいため。

ただし、マネジメントコンソール以外からAmplify Consoleアプリケーションを作る場合は、Gitプロバイダーのトークンを明示的に渡す必要があります。

筆者のチームでは、Secret ManagerにGitHubのPersonal Access Tokenを保存し、それをAWS CDKから参照することで解決しています。

Terraform

2020年9月時点で開発中です。

必要な方はTerraformのIssueにUp Voteすると良いかもしれませんね。

実践Amplify Console

Amplify Consoleを複数のWebアプリおよび本番を含む複数の環境で運用して得られた知見を共有します。

ビルド設定のカスタマイズ

マネジメントコンソール以外から独自のDockerイメージを使用する

ドキュメントには記述がありませんが、マネジメントコンソール以外からでもビルドに使用するイメージを変更できます。

環境変数に \_CUSTOM\_IMAGEを設定し、値にjustincasetech/node:12.14.1-aws-cli-stretch のようにオリジナルのDockerイメージを指定することで可能です。

サービスロールの変更

Amplify ConsoleのCI/CDワークフロー中に、別のAWSリソースにアクセスしたい場合があります。

例えばParameter StoreやSecret ManagerからAPIキーなどを取得するケースでしょうか。

そうした場合、Amplify ConsoleのためのIAM Roleを作成し、Amplifyで使用するように指定するとアクセス可能になります。

なお、サービスロール自体はAmplify Consoleの画面ではなくIAMの画面からでも作成可能です。

その際は Trusted entitiesでamplify.amazonaws.comを指定してください。

メール以外の通知

そもそも、マネジメントコンソールから設定するメール通知は、以下サービスの組み合わせで成り立っています。

Amplify Console → EventBridge → SNS

ここでEventBridgeについて簡単にご説明します。AWSのサービス内で発生した事象を通知してくれるサービスで、例えばECSのタスクの状態変化に応じてLambdaを起動するようなことが可能です。

したがって、Amplify ConsoleのビルドイベントをEventBridgeでフックし、そこからLambdaなどを起動すれば、Slackなど任意の通知が可能になります。

さて、EventBridgeがAmplify Consoleのビルドイベントをフックする仕組みですが、Amplify Consoleアプリケーションと対象のブランチによって一意になる名前のEventBridge Ruleに対して自動でイベントが通知されています。

例えば、アプリケーションのIDがabcde12345678、対象のブランチがmasterなら、amplify-abcde12345678-master-branch-notificationという名前のRuleに対して自動でイベントが通知されます。

ちなみに、全てのブランチが対象の場合はamplify-abcde12345678-AMPLIBRANCHSENTINEL-branch-notificationのような名前になります。

EventBridgeのRuleが定まれば、後はLambdaを定義するだけです。

参考までに、以下のようなイベントが流れてきます。

アイデア集

現場で使えるちょっとしたアイデアです。みなさんのお役に立てば幸いです。

メンテナンス画面の表示

Amplify Consoleでは、カスタムドメインに紐づくブランチを任意のタイミングで切り替えることができます。これを利用して、Webサイトを数秒程度でメンテナンス画面に切り替えられます。

手順は以下の通りです。

1. masterブランチとは別に、maintenanceなどのブランチを作成する。

2. maintenanceブランチでは、Webサイトがメンテナンス中の旨だけを表示するようにする。

3. maintenanceブランチをAmplify Consoleに接続する。

4. メンテナンス中にしたいタイミングで、ドメイン管理から接続先ブランチをmaintenanceにする。

5. メンテナンスが終わったら、ドメイン管理から接続先をmasterに戻す。

ビルド中にスイッチロール

別のAWSアカウントのリソースにアクセスしたい場合などで、ビルド中にスイッチロールを試みたいケースがあると思います。

サービスロールを指定した上で、アクセスしたいAWSアカウントにサービスロールからスイッチできるロールを設定することでスイッチロールができます。

プライベートなGit Submodule

チームによっては、Webアプリケーションで利用しているモジュールをGit Submoduleで管理していると思います。

しかし、2020年9がつ日現在、残念なことにAmplify ConsoleはプライベートなGit Submoduleに対応していません。

したがってGit Subtreeやnpmモジュールを利用したほうが楽なのですが、どうしてもGit Submoduleを利用したい場合にはDeploy KeyをSecret Managerで管理するのがおすすめです。

なお、Amplify Consoleがデフォルトで用意するIAMロールはSecret Managerへのアクセス権限がないので、自前で用意したサービスロールを設定するようにしてください。

トラブルシューティング

筆者がAmplify Consoleを利用していて遭遇したトラブルと、その対応をまとめました。お役に立てば幸いです。

アプリケーションのセットアップに関するトラブル

カスタムドメインのセットアップ時、CNAMEが既に存在するエラー

Amplify ConsoleでホスティングするWebコンテンツは、実際にはAWSが管理するAWSアカウント内のCloudFrontでホスティングされています。

ところで、CloudFrontには、複数のディストリビューションに対して同じCNAMEエイリアスを使用できない制約があります。

したがって、Amplify Consoleに接続したかったCNAMEエイリアスを持つCloudFrontディストリビューションが既にある場合、CNAMEが設定できません。

対策

ドキュメントにあるようにCloudFront側のエイリアス設定を削除すればOKです。

既にCloudFrontでホスティングしているアプリケーションをAmplify Consoleに移行する時などに引っかかるかもしれません。

GitHubリポジトリにWebhookのゴミが残ってしまう

Amplify Consoleでは、AWS側でアプリケーションを削除してもGitHubに登録されたWebhookは削除されません。

お試しでAmplify Consoleの設定と削除を繰り返していると、いつのまにかGitHubリポジトリに大量のWebhookが登録されていることも…

対策

地道に消すしかありません。Amplify ConsoleのアプリケーションIDを控えた上で、GitHubリポジトリのWebhookを一つづつ開き、Payload URLのクエリパラメータを確認して利用中でなければ消します。

Amplify Consoleのアプリケーションを削除するときは、忘れずにWebhookも消すようにしたいですね！

ビルドに関するトラブル

Warning 注意

以下の挙動は2020年9月12日現在、筆者が手元で確認したものです。Amplify Consoleの公式な仕様として公開されているものではありません。

Web Previewのステータスが更新されない

Amplify Console側ではビルドが完了しているにも関わらず、Pull RequestのチェックのステータスがQueued = Waiting to run this check...のまま更新されないことがあります。

短い間隔でCommitを連続してPushした場合、特に直前のCommitに対するWeb Previewのビルドを実行している間に、新しいCommitをPushした場合に起きることがあるようです。

推測ですが、Amplify ConsoleでGitHubのCheckによるビルドが複数走っている場合、初めのビルドが完了した時点で後続のビルドのステータス更新が実行されなくなるのかもしれません。

対策

新しいコミットをPushしましょう。前のコミットのコミットメッセージを更新してforce pushすれば解決します。

または、単にマネジメントコンソールからAmplify Consoleのビルド結果を確認してもよいと思います。

Web Previewが実行されないように見える

Amplify Consoleでは、GitHubのPull Requestに対するWeb Previewの実行中に、新しいWeb Previewが発生するとビルドが保留されます。

保留中のビルドがある場合に更に新しいWeb Previewが発生すると、保留中のビルドが対象とするコミットが書き換わります。その際、GitHub側でコミットに対するチェックが発生していないことになります。

要するに、短い間隔でCommitを3連続以上でPushすると、Pull RequestのCheckからAmplify ConsoleのWeb Previewが消えてしまいます。

対策

ステータスが更新されないケースと同じように、新しいコミットをPushすれば解決します。ただし、保留中のビルドがないことをビルド履歴から確認してください。

ビルドに関するトラブル

GitHubでWeb Previewのステータスが更新されない場合

連続でコミットをPushした場合、2つ目のコミットに対するステータスがQueued = Waiting to run this check...のまま更新されません。2020年9月現在、以下の手順で再現します。

1. GitHubで特定の Pull Request において、コミットを Push する (コミットA)

2. コミットA のビルドの実行中に、コミットを Push する (コミットB)

3. コミットA とコミットB 共にビルドは完了する

4. GitHub の Pull requests -> Commit においてコミットA は "All checks have passed" と表示されるが、コミットB は "Some checks haven’t completed yet" と表示され続ける

対策

新しいコミットをPushしましょう。前のコミットのコミットメッセージを更新してforce pushすれば解決します。

または、単にマネジメントコンソールからAmplify Consoleのビルド結果を確認してもよいと思います。

GitHubでWeb Previewが実行されない場合

連続でコミットをPushした場合、中間のコミットに対するビルドが省略されます。

Amplify Consoleでは、GitHubのPull Requestに対するWeb Previewの実行中に、新しいWeb Previewが発生するとビルドが保留されます。

保留中のビルドがある場合に更に新しいWeb Previewが発生すると、保留中のビルドが対象とするコミットが書き換わります。その際、GitHub側でコミットに対するチェックが発生していないことになります。

対策

ステータスが更新されないケースと同じように、新しいコミットをPushすれば解決します。ただし、保留中のビルドがないことをビルド履歴から確認してください。

GitHubでPull RequestのTarget Branch変更後、Web Previewが有効にならない場合

GitHubではPull RequestのTarget Branchが別のBranch(例えばmaster)にmergeされた場合、そのPull RequestのTarget Branchが自動的にmergeされたBranch（master）に変更されます。

しかし、その場合にAmplify ConsoleのWeb Previewが自動的に有効にならないケースがあります。以下の手順で再現します。

1. あるGitHubリポジトリで、masterに対するWeb Previewのみを有効化

2. master をもとに branch\_a を作成

3. master <= branch\_a 方向の PR (#1) を作成

4. branch\_a をもとに branch\_b を作成

5. branch\_a <= branch\_b 方向の PR (#2) を作成

6. PR #1 をマージし、branch\_a を削除

7. PR #2 が master <= branch\_b となるが、Amplify Console 側の Web Preview では PR が作られない

対策

一度Pull Requestをクローズし、新たに作成するしかありません。

マネジメントコンソールにビルドのステータスが表示されない

マネジメントコンソールでAmplify Consoleを日本語で表示すると、日本語のビルド・デプロイ・検証に加えて英語のBuild・Deploy・Verifyタブも表示されてしまうバグがあります。

タブが重複して表示されているときは、英語を無視して日本語側を開くと期待している結果が得られます。

巨人の肩の上に乗る

本書の執筆にあたって参考にさせていただいたドキュメント・ブログをご紹介します。本書のトピックについて理解を深めるために、ぜひご活用下さい。

AWS公式ドキュメント

Amplify Consoleの開発者向けドキュメントです。

ブログ・LT

Amplify Console 誕生以来本番運用しつづけてわかったこと #awswakaran\_tokyo

2018年のリリース以来Amplify Consoleを本場運用している方の知見がまとまったスライドです。NetlifyやFirebase Hostingとの比較もあります。

Amplify Console のビルド通知をSlackで受け取るためにやったこと}

本書のレビューもしていただいた@youta1119さんのAmplify Meetup登壇資料です。EventBridgeの挙動について詳しいです。

12 New Features of AWS Amplify

2020年1月のブログで、Amplifyのアップデートを取り上げています。特に即時キャッシュ無効化・差分デプロイ・Webhook・マニュアルデプロイについて簡潔にまとまっています。

あとがき

ここまで読んでいただき、どうもありがとうございました。

Amplify Consoleの導入を検討されている方が、少しでも具体的なイメージを持てれば幸いです。また、既にAmplify Console を実践されている方にとっても、多少なりとも参考になれば嬉しいです。

本書の執筆にあたっては情報の正確性を期すための努力を行っていますが、不正確な表現などがあればすべて筆者の責任です。

そうした点については、今後の版で随時更新させていただきたいのでお知らせいただけ

ると幸いです。

謝辞

執筆にあたっては、justInCaseで一緒に（というか、リードして）Amplify Consoleの導入を進めていただいた@youta1119さんにレビューをいただきました。

また、ドラフトの段階で第一線でAmplify Consoleの導入を進められている匿名希望のエンジニアの方にアドバイスをいただきました。筆者が活用しきれていない機能やユースケースをご紹介いただき、大変勉強になりました。

その他、事前に感想を寄せてくださった皆様のおかげで読みやすくできました。

お忙しい中、皆様どうもありがとうございました。

著者紹介

金融系のスタートアップでアプリケーション・インフラを担当しています。

Amplify Consoleの導入前はフロントエンドエンジニアと頑張ってCloudFront + S3を構築していました。Amplify Console のなんとラクなことか！

今後AWSでフロントエンド構築するなら第一の選択肢になると思う反面、検索するとAmplify CLIのことばかり出てきて悲しくなったので本書を書きました。

本書のご意見やご感想など、ぜひツイートいただけると嬉しいです。もっとAmplify Console事例を知りたいので、ぜひ何でも聞きたいです！

Mail: hiroga1030@gmail.com

Twitter: @hiroga\_cc