

# AWS Builders Online Series

T2-4

## まず知っておきたい！ サーバー開始と停止の自動化による コスト削減の第一歩

佐藤 裕介

アマゾン ウェブ サービス ジャパン合同会社  
技術統括本部 インダストリソリューション 第一部

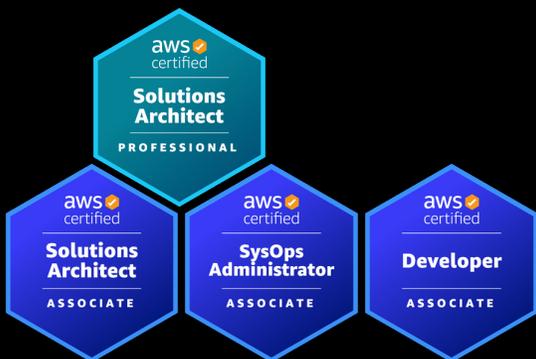
山澤 良介

アマゾン ウェブ サービス ジャパン合同会社  
ソリューションアーキテクト

# 自己紹介

## ソリューションアーキテクト 佐藤 裕介

これからAWSを使い始める、または、使い始めた製造業のお客様を中心に支援をしています



## ソリューションアーキテクト 山澤 良介

ソリューションアーキテクトとして、業種業態問わず様々なお客様を支援しています。



# 本日本話すること、しないこと

## 本日本話すること

- AWSソリューションのご紹介
- サーバーを時刻設定で開始、停止するソリューション

## 本日本話しないこと

- Amazon EC2、Amazon RDSといった利用されているAWSサービスの詳細
- 紹介するソリューションであるInstance Scheduler on AWSの設定方法

# Agenda

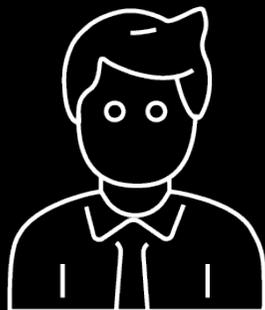
- すぐに使える！AWSソリューションとは
- サーバーを開始、停止するためには？
- Instance Scheduler on AWSのご紹介
- デモ
- まとめ



# すぐに使える！ AWSソリューションとは



# こんな経験はありませんか？



解決したい課題があるけど、  
すでに誰かがやっていて解決策どこかに公開されていないかな・・・

使っていない時間にサーバー停止したいんだけど、誰か同じことやってないかなあ  
「AWS サーバー 自動停止」で調べてみよう

# AWSソリューションライブラリとは

- AWSは世界中のお客様にご利用されているため、同じ課題を持ったお客様が世界のどこかにおり、すでに解決している可能性がある
- すでに構築された解決策や参考となるアーキテクチャを再利用することで、ビジネスのスピードを加速することができる
- 業界別（自動車、消費財、ゲームなど）やテクノロジー別（分析、機械学習、セキュリティなど）で、**構築済みのソリューション**や参考となるアーキテクチャ図を公開している

## AWS ソリューションライブラリ

ビジネスおよび技術的なユースケースのための検証済みのソリューションとガイダンス

AWS Innovate 登録開始！クラウドをフル活用したアプリ開発を集中して学ぶ 1日 - 10/20 (木) 開催

今すぐ登録 >

### AWS および AWS パートナーのソリューションについて知る

今日の組織は、ビジネス上の課題を迅速に解決するための検証済みのソリューションとアーキテクチャガイダンスを求めています。お客様が既製のデプロイをご希望の場合でも、カスタマイズ可能なアーキテクチャをご希望の場合でもご安心ください。AWS ソリューションライブラリは、幅広い業界およびテクノロジーのユースケース向けに AWS および AWS パートナーによって構築されたソリューションを提供します。

**AWS のサービス**  
専用のクラウド製品

**AWS ソリューション**  
AWS のサービス、コード、設定を組み合わせたすぐにデプロイできるソリューション

**パートナーソリューション**  
AWS パートナーのソフトウェア、SaaS、マネージドサービス

**ガイダンス**  
規範的なアーキテクチャ図、サンプルコード、技術コンテンツ

# AWSソリューションとは

- 多くのお客様で発生した共通した課題に対するソリューションを提供
  - 例えば、「使っていない時間帯にサーバーを停止したい」
  - 例えば、「ログを集約して検索出来るようにしたい」
- 2022年12月時点で全83個のソリューションを公開中



The screenshot displays the AWS Solutions Center interface. At the top, a dark blue banner reads "すべてのテクノロジーのソリューションを参照" (View all solutions for all technologies). Below this, a blue navigation bar contains the text "今日から AWS を始めましょう | 簡単 10 分間チュートリアルやユースケース別構築ガイドで AWS の開始方法を確認" (Start with AWS today | Check the starting methods of AWS with simple 10-minute tutorials or use case-specific building guides) and a "詳細を見る >>" (View details >>) button.

The main content area features a search bar with the text "AWS ソリューションを検索" (Search for AWS solutions). Below the search bar, it indicates "1-30 (83)" results and a sorting option "並べ替え: 名前 (A->Z)" (Sort by: Name (A->Z)).

On the left side, there are two filter sections:

- ソリューション別に絞り込む:** (Filter by solution type)
  - すべてのフィルターをクリア (Clear all filters)
  - コンテンツタイプ (Content type)
    - AWS ソリューション
    - AWS パートナーソリューション
    - ガイダンス
  - テクノロジーカテゴリ\* (Technology category)
    - 分析
    - アプリケーション開発 & DevOps
    - ブロックチェーン

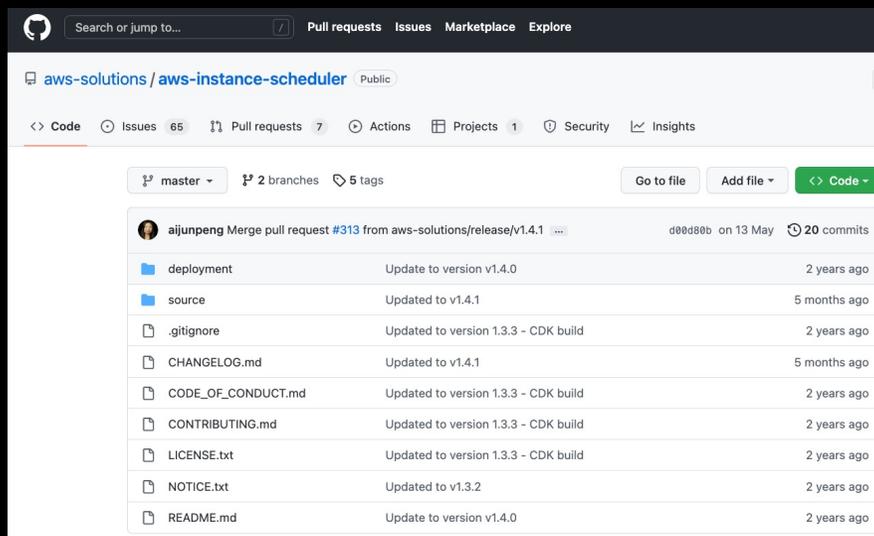
The main content area displays three solution cards:

- AWS CloudEndure Migration Factory ソリ...** (Updated) - AWS により構築 (Built by AWS). Description: 手動で行っていた多くの移行プロセスの調整と自動化を行うソリューションをデプロイし、CloudEndure Migration を活用しながら高度に自動化されたリフト
- AWS Connected Mobility Solution** (New) - AWS により構築 (Built by AWS). Description: サーバーレス IoT アプリケーションを構築して、クラウドで車両やマイクロモビリティからのデバイスデータのライフサイクルを安全に管理します。
- AWS Content Analysis** (Updated) - AWS により構築 (Built by AWS). Description: 機械学習が生成するメタデータを通じてメディアファイルを分析し、有意義なインサイトを生成するために、AWS の AI サービスを使用するソリューションをデプロイします。

# 具体的には何が提供されているのか

- すでに組み合わされたAWSサービスをデプロイするCloudFormationテンプレート
- GitHubにて公開されたソースコード
- 使い方が書かれたデプロイガイド

```
1 {
2   "Description": "(S00030) The AWS CloudFormation template for
3   deployment of the aws-instance-scheduler, version: v1.4.1",
4   "AWSTemplateFormatVersion": "2010-09-09",
5   "Metadata": {
6     "AWS::CloudFormation::Interface": {
7       "ParameterGroups": [
8         {
9           "Label": {
10            "default": "Scheduler (version v1.4.1)"
11          },
12          "Parameters": [
13            "TagName",
14            "ScheduledServices",
15            "ScheduleRdsClusters",
16            "CreateRdsSnapshot",
17            "SchedulingActive",
```



CloudFormation  
テンプレート

ソースコード

デプロイガイド

<https://aws.amazon.com/jp/solutions/implementations/instance-scheduler>

# CloudFormationテンプレートとは

- AWS CloudFormationは、何をどう作りたいかをコードで記述することでデプロイを自動化するサービス
- CloudFormationテンプレートは、何をどう作りたいかを記述したコード自体



CloudFormation  
テンプレート



AWS CloudFormation

```
AWSTemplateFormatVersion: '2010-09-09'
```

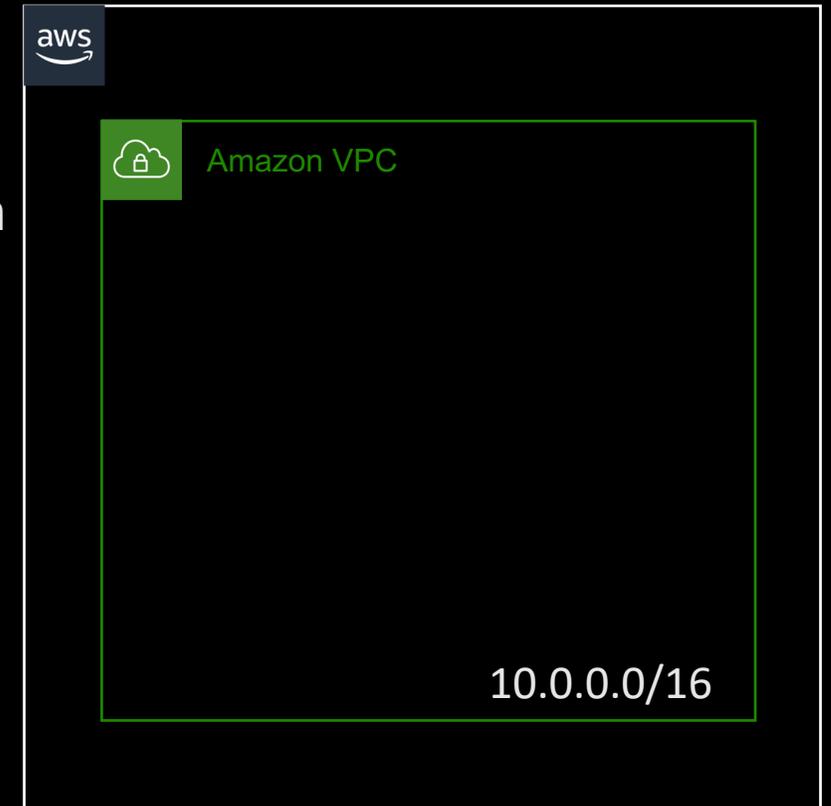
```
Resources:
```

```
  FirstVPC:
```

```
    Type: AWS::EC2::VPC
```

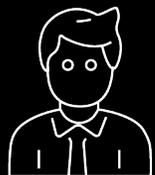
```
    Properties:
```

```
      CidrBlock: 10.0.0.0/16
```

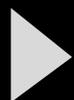


「Amazon VPCを10.0.0.0/16で作ってください」

# AWSソリューションの一例



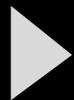
使っていない時間もサーバーが起動して無駄な気がします



## Instance Scheduler on AWS

起動と停止のスケジュールを定義して、サーバーを自動的に停止、再開できます

複数のアカウントやリージョンのログを集約して検索したいです



## Centralized Logging on AWS

複数のアカウントやリージョンのCloudWatch Logsを収集、分析、可視化できます

負荷テスト用に負荷を掛けるためのリクエストを送るサーバーを用意するのが大変です



## Distributed Load Testing on AWS

AWSのコンテナ実行基盤を利用して、負荷テスト用のリクエストを送付できます

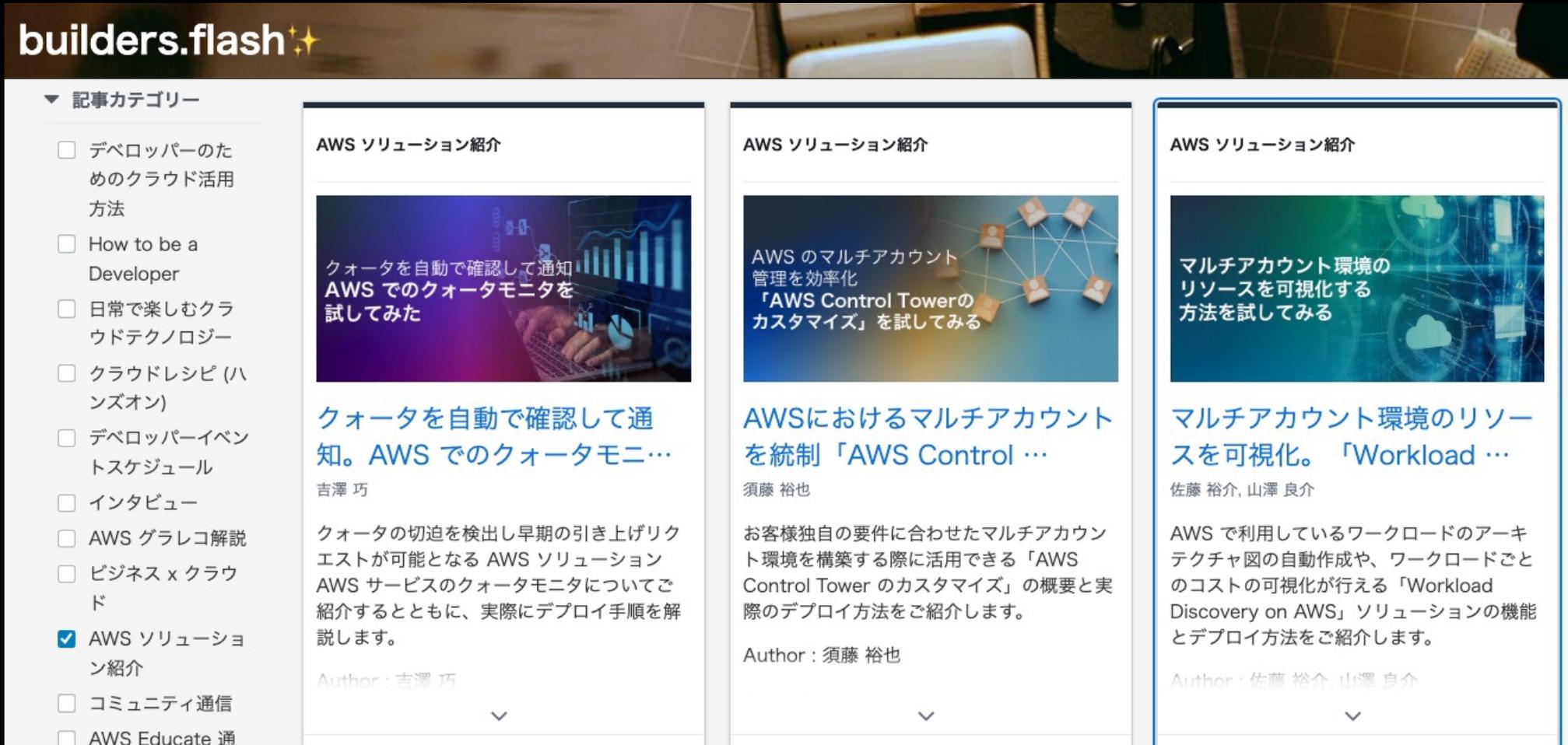
同じことを誰かが悩んでいそうだなと思ったら・・・まずは検索！

AWSソリューション 負荷テスト



# AWSソリューションを紹介する連載もあります

builders.flashというAWSのウェブマガジンでAWSソリューションを紹介する連載も行っています



**builders.flash** ✨

▼ 記事カテゴリー

- デベロッパーのためのクラウド活用方法
- How to be a Developer
- 日常で楽しむクラウドテクノロジー
- クラウドレシピ (ハンズオン)
- デベロッパーイベントスケジュール
- インタビュー
- AWS グラレコ解説
- ビジネス x クラウド
- AWS ソリューション紹介
- コミュニティ通信
- AWS Educate 通

### AWS ソリューション紹介



クォータを自動で確認して通知  
AWS でのクォータモニタを試してみた

#### クォータを自動で確認して通知。AWS でのクォータモニ...

吉澤 巧

クォータの切迫を検出し早期の引き上げリクエストが可能となる AWS ソリューション AWS サービスのクォータモニタについてご紹介するとともに、実際にデプロイ手順を解説します。

Author : 吉澤 巧

### AWS ソリューション紹介



AWS のマルチアカウント管理を効率化  
「AWS Control Towerのカスタマイズ」を試してみる

#### AWSにおけるマルチアカウントを統制「AWS Control ...

須藤 裕也

お客様独自の要件に合わせたマルチアカウント環境を構築する際に活用できる「AWS Control Towerのカスタマイズ」の概要と実際のデプロイ方法をご紹介します。

Author : 須藤 裕也

### AWS ソリューション紹介



マルチアカウント環境のリソースを可視化する  
方法を試してみる

#### マルチアカウント環境のリソースを可視化。「Workload ...

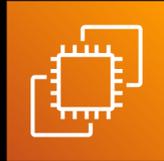
佐藤 裕介, 山澤 良介

AWS で利用しているワークロードのアーキテクチャ図の自動作成や、ワークロードごとのコストの可視化が行える「Workload Discovery on AWS」ソリューションの機能とデプロイ方法をご紹介します。

Author : 佐藤 裕介, 山澤 良介

サーバーを開始、停止する  
ためには？

# 本ソリューションの対象となるサービス



## Amazon EC2

500以上の種類から選択出来る  
仮想サーバーのサービス

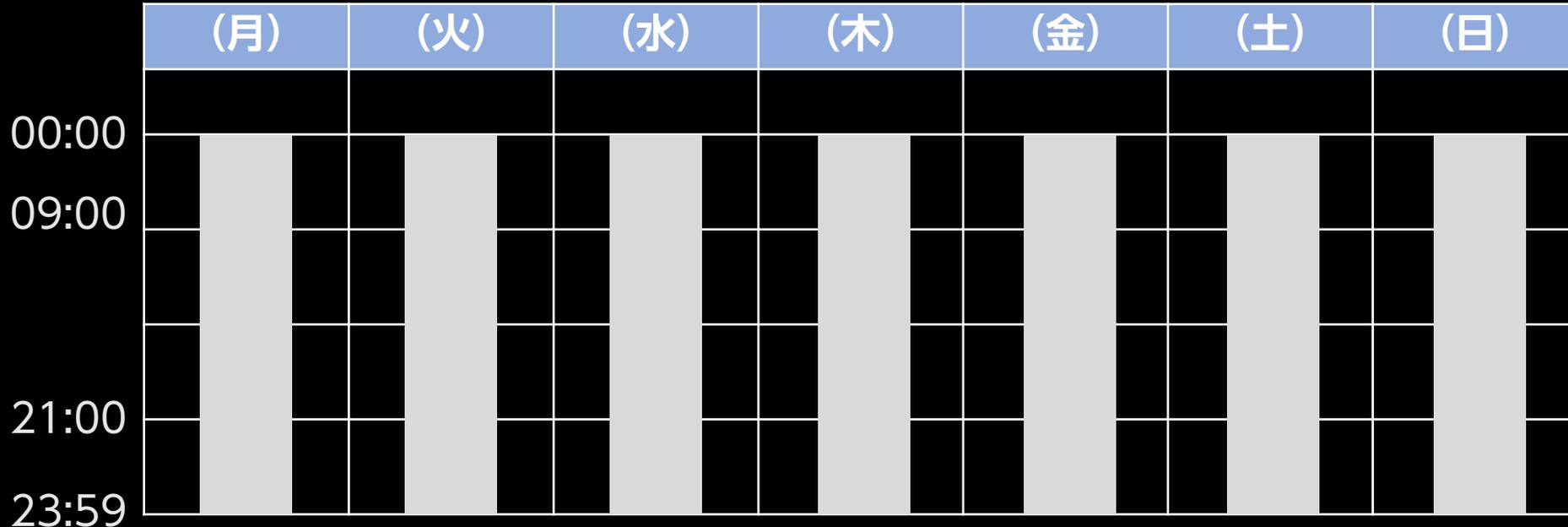


## Amazon RDS

MySQL, PostgreSQL, Oracle, SQL Serverなどを  
実行できるデータベースのサービス

# そのサーバーは常に起動している必要がありますか？

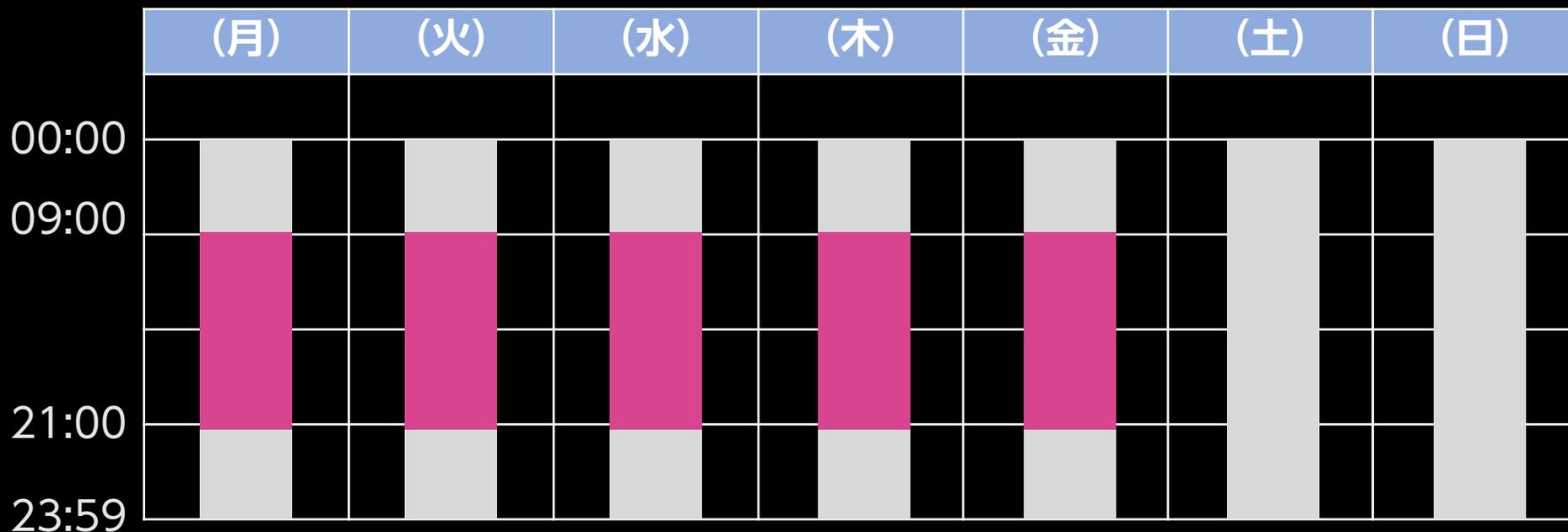
Amazon EC2を24時間365日起動している場合



1時間あたり \$ 0.248 × 24時間 × 30日 = ひと月あたり **\$ 178.56**

# そのサーバーは常に起動している必要がありますか？

開発用のAmazon EC2なので、平日9:00～21:00しか使っていない

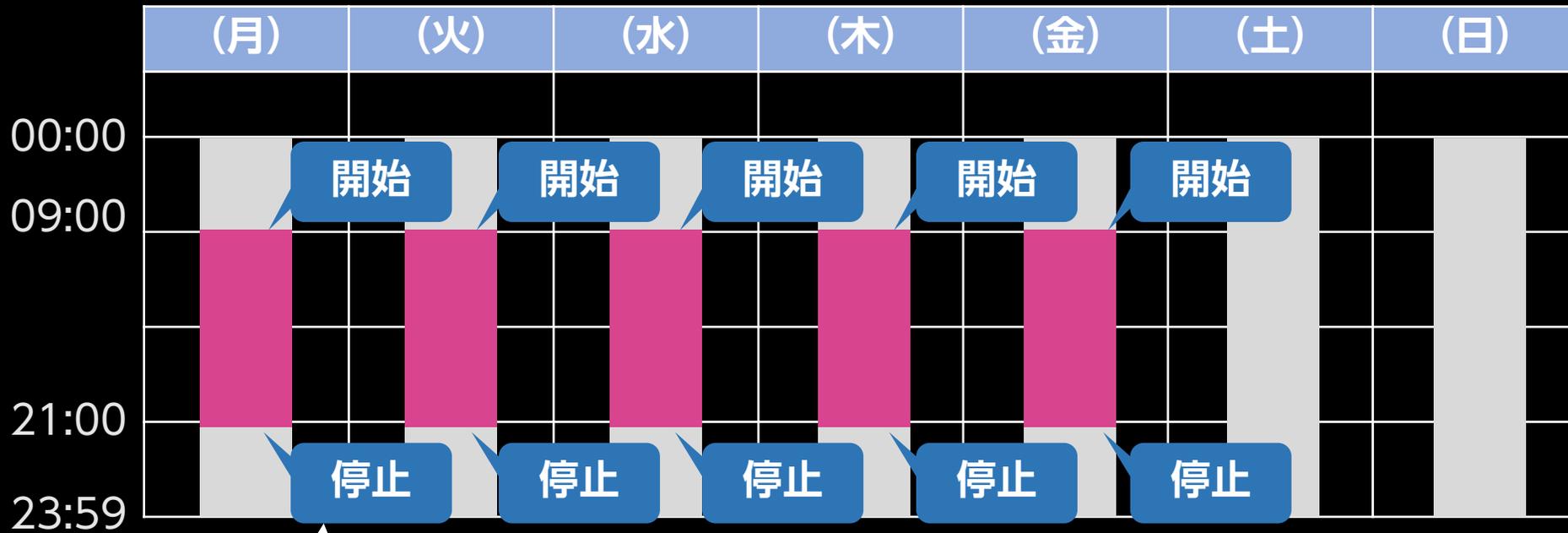


1時間あたり \$0.248 × **12** 時間 × **22** 日 = ひと月あたり **\$ 65.47 (64%減)**

24時間起動と比較した場合  
東京リージョン、Linux, m6i.xlargeにて試算

# 使い始めるときに開始、使い終わったら停止

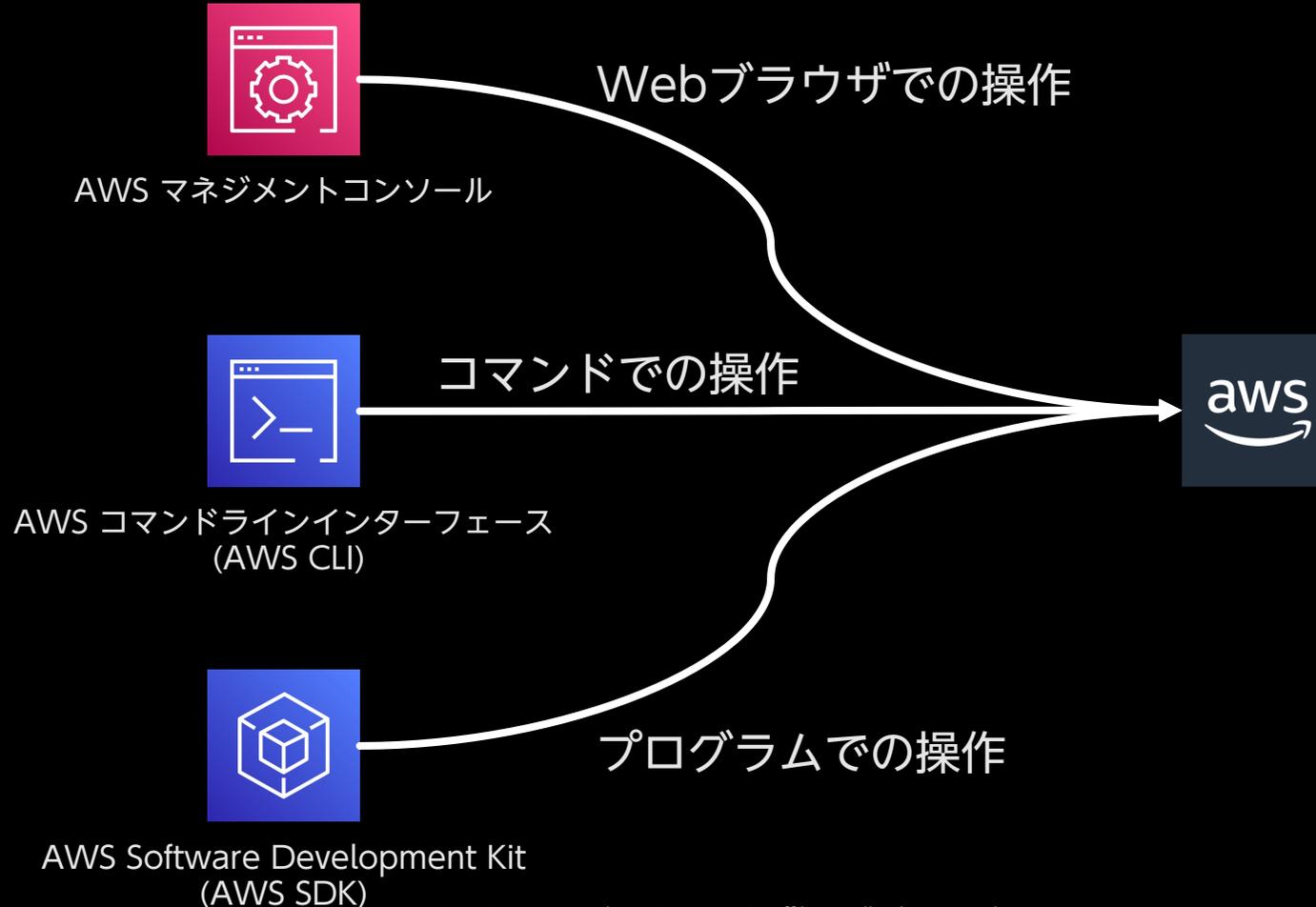
Amazon EC2及びAmazon RDSはサーバーレスのサービスではないため、開始と停止を指示する必要がある



「09:00に開始」  
「21:00に停止」

# AWSの操作方法

AWSはマネジメントコンソール、CLI、SDKによりAWSのAPIを呼び出すことで操作出来る



# マネジメントコンソールによる停止再開の方法



マネジメントコンソールから「インスタンスを停止」「インスタンスを開始」

AWS マネジメントコンソール



GUI（グラフィカルユーザーインターフェース）のため簡単だが、  
手作業のため、停止し忘れ、起動し忘れ、オペレーションミスが発生する可能性がある

# AWS CLIによる停止再開の方法



AWS コマンドライン  
インターフェース (AWS CLI)

ターミナルから「インスタンスを停止 (stop-instance) 」  
「インスタンスを開始 (start-instance) 」

A screenshot of a terminal window with a dark background and light text. The window has three colored window control buttons (red, yellow, green) at the top left. The prompt is a tilde (~). The command entered is 'aws ec2 stop-instances --instance-ids i-XXXXXXXXXX' followed by a cursor. To the left of the terminal window is a white line-art icon of a person sitting at a laptop.

```
~  
> aws ec2 stop-instances --instance-ids i-XXXXXXXXXX
```

複数のインスタンスを指定したり、定期的に行うための仕組みが  
別途必要になる。また、コマンドを実行する環境が必要となる。

# AWS SDKによる停止再開の方法



AWS Software Development Kit  
(AWS SDK)

プログラムから「インスタンスを停止 `start_instances()`」  
「インスタンスを開始 `stop_instances()`」

stop-instances.py (Pythonで記述した例)

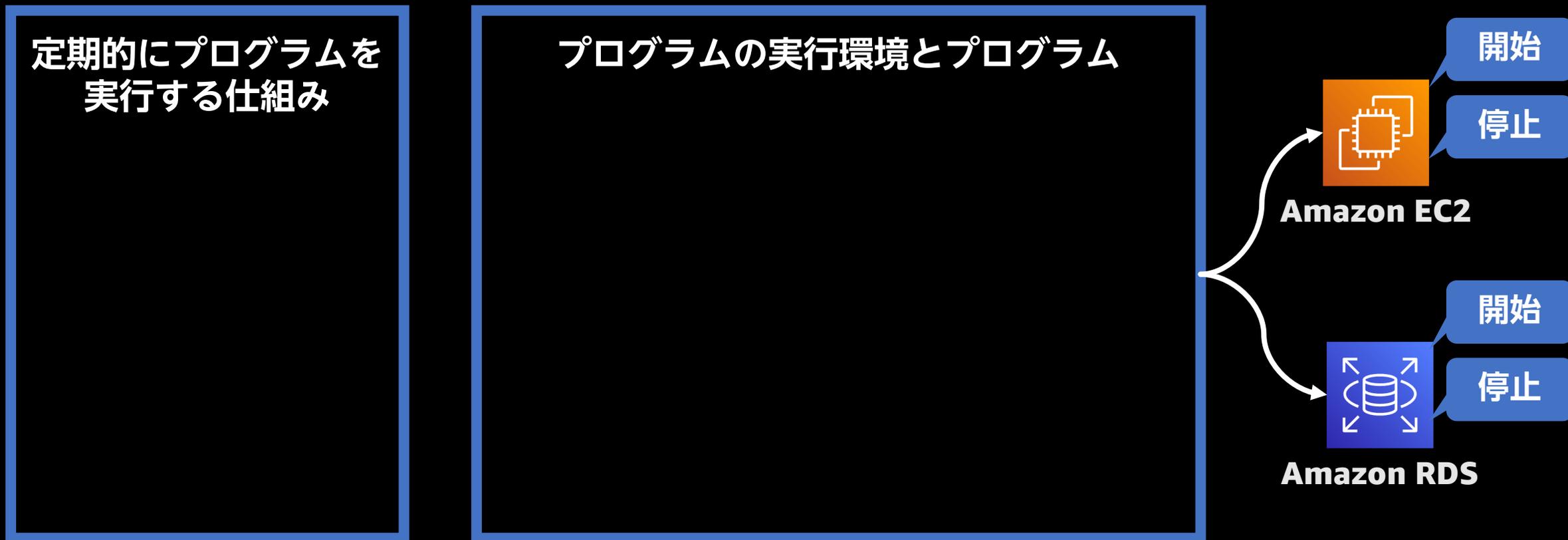
```
import boto3

client = boto3.client('ec2')
response = client.stop_instances(InstanceIds='i-XXXXXXXXXXXX')
```

プログラムを記述し、自身で開発する必要がある。  
定期的に行うための仕組みが別途必要になる。  
また、コードを実行する環境が必要となる。

# サーバーを自動で停止、開始するには・・・

定期的にプログラムを実行する仕組み、プログラムの実行環境、動かすプログラムが必要

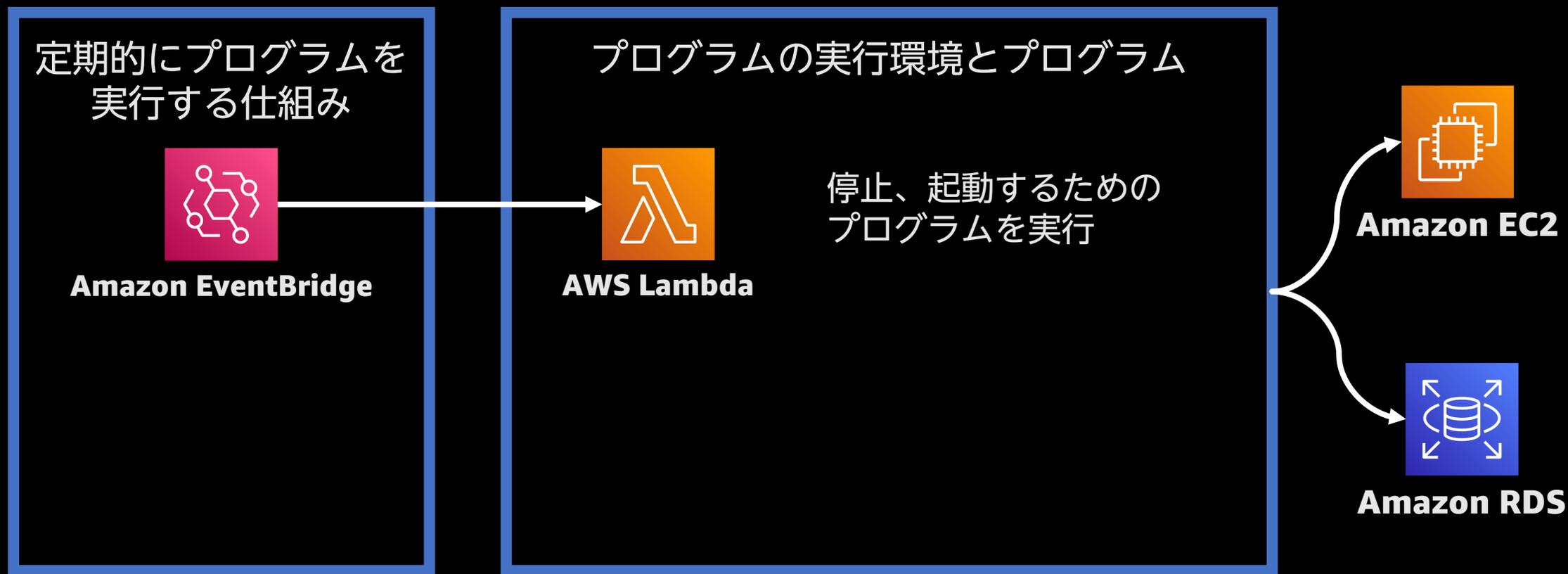


# Instance Scheduler on AWS のご紹介



# AWSソリューションを活用した停止再開の方法

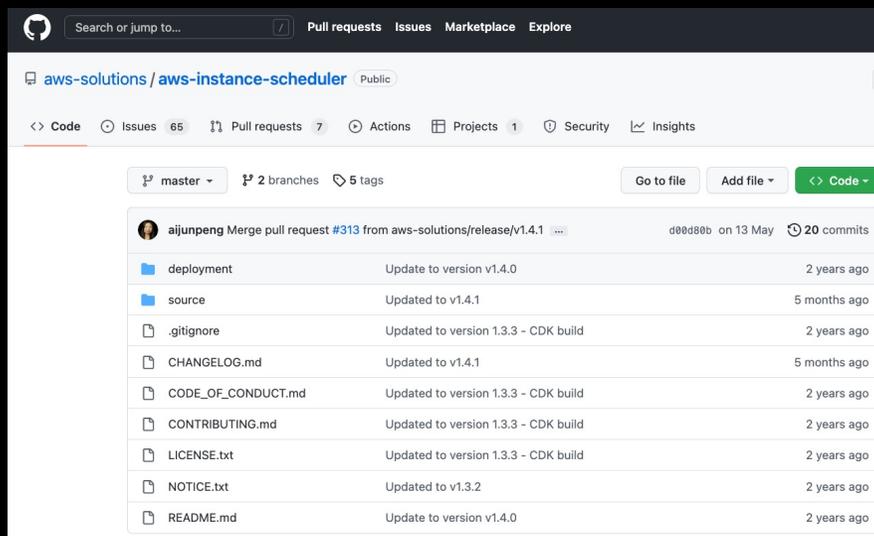
定期的にプログラムを実行する仕組みと、プログラムの実行環境はAWSのサービスとして提供されている



# 具体的には何が提供されているのか

- すでに**組み合わせられたAWSサービス**をデプロイするCloudFormationテンプレート
- GitHubにて**公開されたソースコード**
- 使い方が書かれた**デプロイガイド**

```
1 {
2   "Description": "(S00030) The AWS CloudFormation template for
3   deployment of the aws-instance-scheduler, version: v1.4.1",
4   "AWSTemplateFormatVersion": "2010-09-09",
5   "Metadata": {
6     "AWS::CloudFormation::Interface": {
7       "ParameterGroups": [
8         {
9           "Label": {
10            "default": "Scheduler (version v1.4.1)"
11          },
12          "Parameters": [
13            "TagName",
14            "ScheduledServices",
15            "ScheduleRdsClusters",
16            "CreateRdsSnapshot",
17            "SchedulingActive",
18            "..."
19          ]
20        }
21      ]
22    }
23  }
```



CloudFormation  
テンプレート



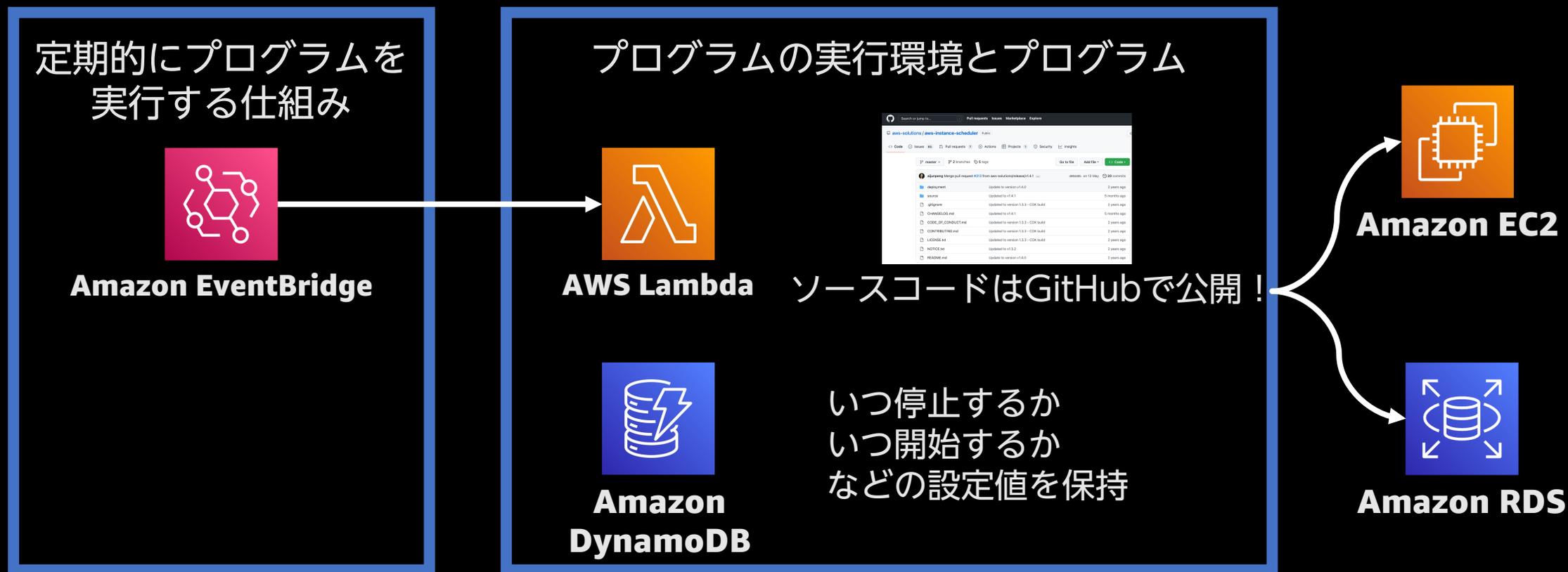
ソースコード

デプロイガイド

<https://aws.amazon.com/jp/solutions/implementations/instance-scheduler>

# Instance Scheduler on AWS (概略図)

公開されているCloudFormationテンプレートとソースコードを利用し、自分で開発することなくサーバーの自動的な開始や停止を実現出来る

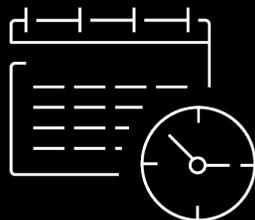


# Instance Scheduler on AWSの利点



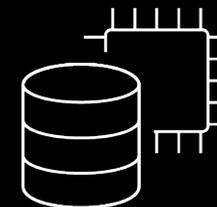
## 簡単にデプロイ

CloudFormationテンプレート、  
ソースコードは公開されているため、  
自身でコーディングが不要



## 柔軟なスケジュール設定

日を跨ぐスケジュール、曜日指定、  
毎月第一月曜日といった柔軟な  
スケジュール設定が分単位で可能



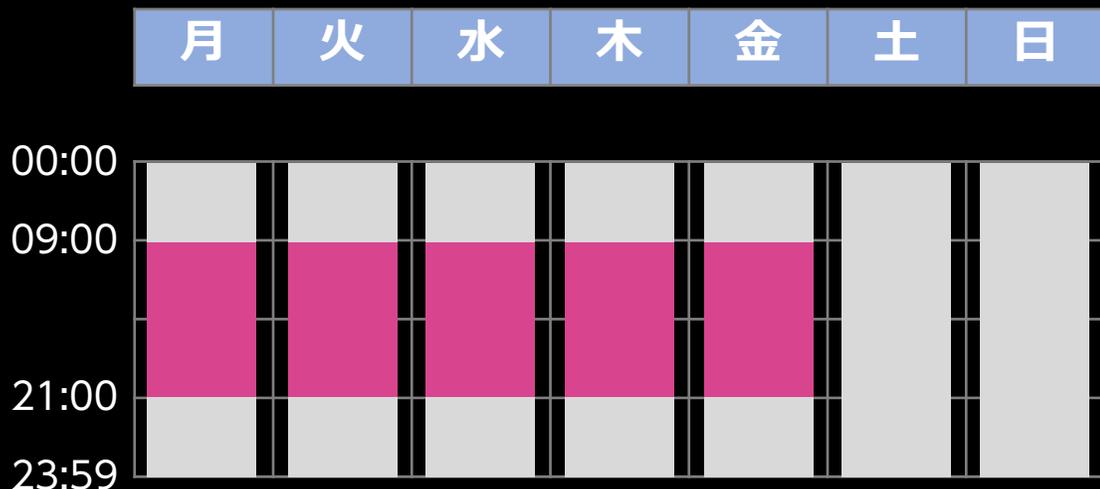
## 複数リソースにも対応

リソースタグにより対象を指定するため、  
複数の Amazon EC2, Amazon RDS に  
簡単にスケジュールを適用可能

# 柔軟なスケジュール設定

開始、停止したい曜日、時刻でピリオド定義を作成

平日09:00～21:00まで起動、それ以外は停止



## ピリオド定義

フィールド名	値
name	mon-fri-9am-9pm
begintime	09:00
endtime	21:00
weekdays	mon-fri



Amazon  
DynamoDB

## ピリオド定義

識別する名前、インスタンスの開始時刻、終了時刻、曜日を設定

# 柔軟なスケジュール設定

作成したピリオド定義からスケジュール定義を作成し、  
停止・開始したいインスタンスにリソースタグで紐付け

## ピリオド定義

フィールド名	値
name	mon-fri-9am-9pm
begintime	09:00
endtime	21:00
weekdays	mon-fri

## スケジュール定義

フィールド名	値
name	Japan-office-hours
periods	mon-fri-9am-9pm
timezone	Asia/Tokyo



## スケジュール定義

タイムゾーンを指定し、適用したいピリオド定義の名前を設定する  
開始・停止したいインスタンスにはリソースタグで、スケジュール定義の名前を指定する

# 複数リソースにも対応

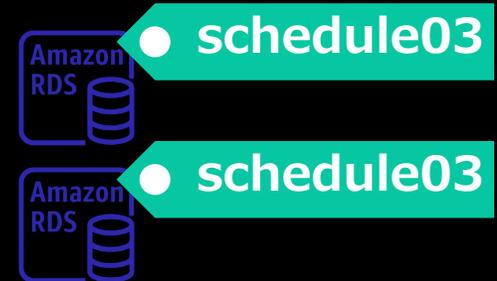
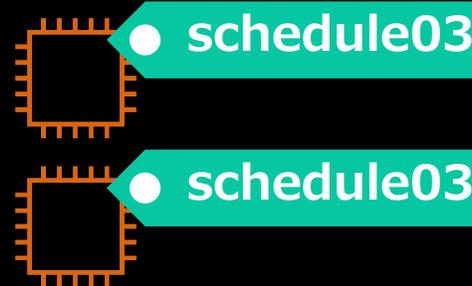
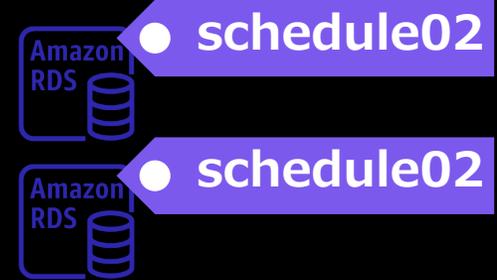
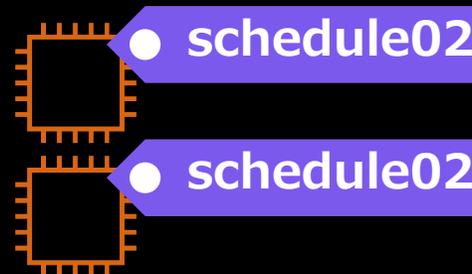
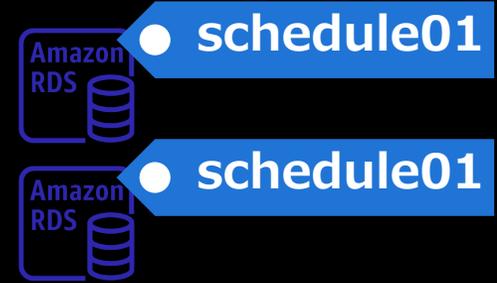
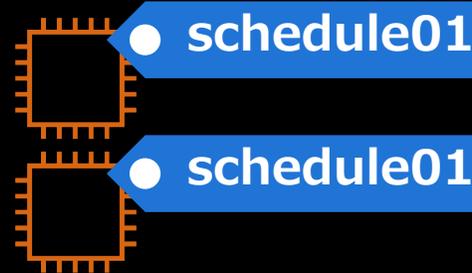
リソースタグで識別するため、複数のリソースにも簡単に対応可能

## スケジュール定義

フィールド名	値
name	schedule01
periods	mon-fri-9am-9pm
timezone	Asia/Tokyo

フィールド名	値
name	schedule02
periods	every-sat-sun
timezone	Asia/Tokyo

フィールド名	値
name	schedule03
periods	every-mon
timezone	Asia/Tokyo



# まとめ



# まとめ

- AWSは世界中で利用されており、AWSソリューションとしてよくある課題の解決策がすでに公開されている可能性がある
- AWSソリューションはAWSサービスの構築、プログラムの開発が不要ですぐに利用できる
- Amazon EC2、Amazon RDSを指定した時刻で開始、停止する方法として Instance Scheduler on AWSをご紹介します
- Instance Scheduler on AWSや他のAWSソリューションは日本語で紹介ブログを書いています  
サーバーを指定した期間で停止する「AWS Instance Scheduler」を試してみる - builders.flash☆

<https://aws.amazon.com/jp/builders-flash/202110/instance-scheduler/>



# 参考リンク

サーバーを指定した期間で停止する「AWS Instance Scheduler」を試してみる

[https://aws.amazon.com/jp/builders-flash/202110/instance-scheduler/?awsf.filter-name=\\*all](https://aws.amazon.com/jp/builders-flash/202110/instance-scheduler/?awsf.filter-name=*all)

## Instance Scheduler on AWS

<https://aws.amazon.com/jp/solutions/implementations/instance-scheduler/>

## デプロイガイド（日本語）

[https://d1.awsstatic.com/Solutions/ja\\_JP/instance-scheduler-on-aws.pdf](https://d1.awsstatic.com/Solutions/ja_JP/instance-scheduler-on-aws.pdf)

[https://docs.aws.amazon.com/solutions/latest/instance-scheduler-on-aws/welcome.html?refid=sl\\_card](https://docs.aws.amazon.com/solutions/latest/instance-scheduler-on-aws/welcome.html?refid=sl_card)

## GitHub

<https://github.com/aws-solutions/aws-instance-scheduler>

# AWS TRAINING & CERTIFICATION

## AWS Skill Builder の 500+ の 無料デジタルコースで学ぼう

30以上のAWSソリューションの中から、自分に最も関係のあるクラウドスキルとサービスにフォーカスし、自習用のデジタル学習プランとRamp-Upガイドで学ぶことができます。

- 自分のペースでAWSクラウド上を活用した未来を切り開く
- 学習プランでスキルや知識を向上
- AWS認定資格でクラウドの専門知識を証明する

自分に合ったスキルアップ方法で学びましょう  
[EXPLORE.SKILLBUILDER.AWS](https://explore.skillbuilder.aws) »



# AWS Builders Online Series に ご参加いただきありがとうございます

楽しんでいただけましたか? ぜひアンケートにご協力ください。  
本日のイベントに関するご意見/ご感想や今後のイベントについてのご希望や改善のご提案などがございましたら、ぜひお聞かせください。



[aws-apj-marketing@amazon.com](mailto:aws-apj-marketing@amazon.com)



[twitter.com/awscloud\\_jp](https://twitter.com/awscloud_jp)



[facebook.com/600986860012140](https://facebook.com/600986860012140)



<https://www.youtube.com/user/AmazonWebServicesJP>



<https://www.linkedin.com/showcase/aws-careers/>



[twitch.tv/aws](https://twitch.tv/aws)

# Thank you!

Sato Yusuke  
Solutions Architect

Yamazawa Ryosuke  
Solutions Architect