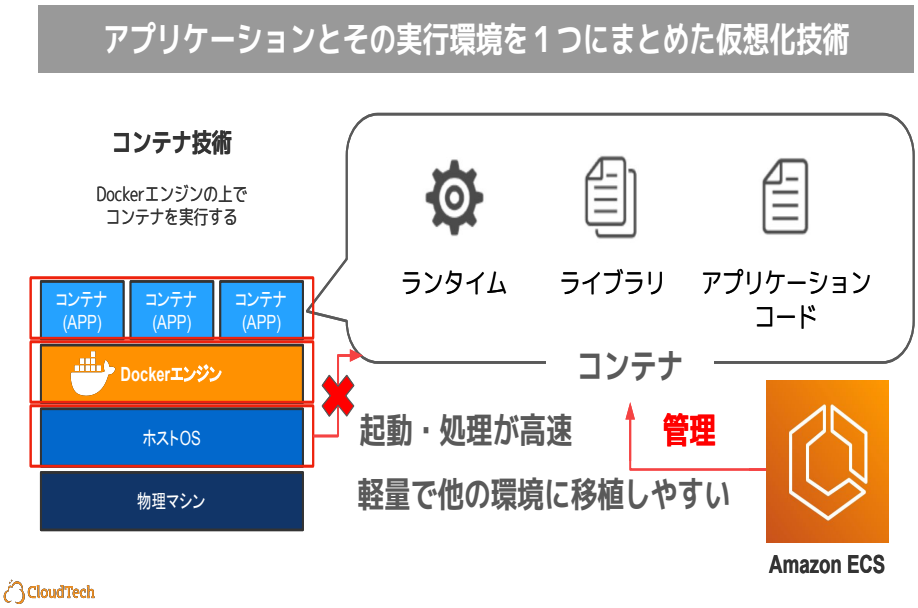
Amazon ECSについて解説します。

コンテナとは？



まず、ECSを学ぶ前にコンテナとは何かを押さえておくことが重要です！

コンテナとは**アプリケーションとその実行環境を1つにまとめた仮想化技術、またはその技術を使って立ち上げたアプリケーションのこと**を指し**このような図**で表現することができます。
Dockerエンジンと呼ばれるコンテナを動かすためのソフトウェアをインストールし、**コンテナ**を構築します。

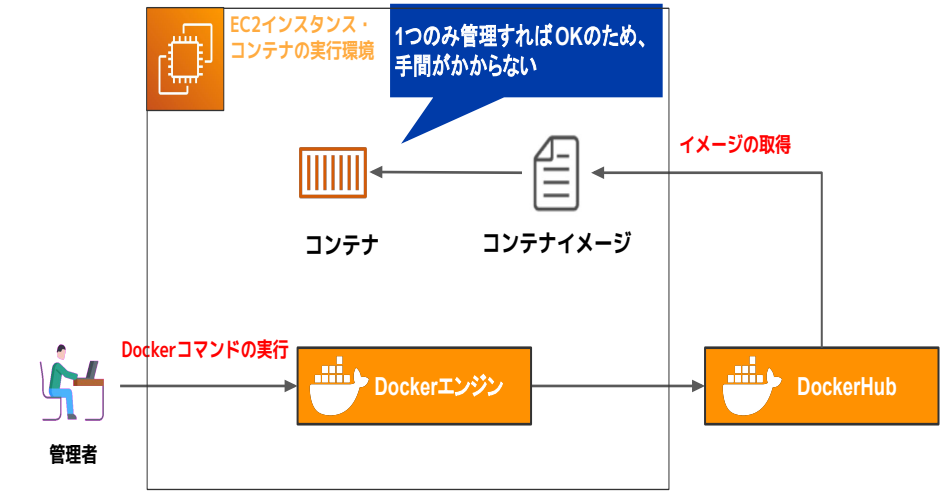
コンテナの中に含まれているものは、アプリを動かすために必要な**ランタイム・依存ライブラリ・アプリケーションコード**などです。
プログラミング言語で書いたコードと、そのコードを動かすために必要な部品がひとまとめになっているイメージです。

OSは含まれてないことがポイントです。
OSを含んでいないため、**起動時間や処理を高速**にできたり、**軽量であり、他の環境に移行しやすい**というメリットがあります。

ここでは簡単に概要だけお伝えしましたが、よりコンテナについて詳しく知りたい方は「【Docker講座】」をぜひご視聴ください。

さて、このコンテナを管理するサービスが今回学ぶ **Amazon ECS**となります。
では、なぜそもそもコンテナの管理が必要なのかみてみましょう！

コンテナ管理



CloudTech

まず動作しているコンテナが 1つのみの場合をみてみましょう。
今回は**EC2インスタンス**がコンテナの実行環境となっている場合を想定します。

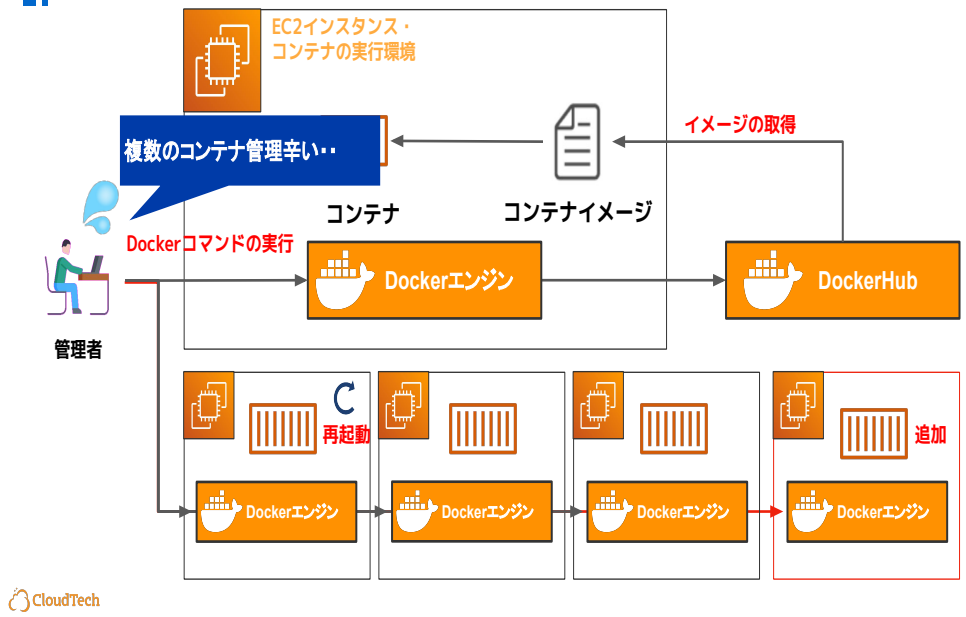
管理者はコンテナを動かすために**Dockerエンジン**に**Dockerコマンド**と呼ばれるコマンドを実行します。
すると、コンテナのイメージを保管している **Docker Hubと呼ばれるサービス**から**コンテナのイメージを取得**します。
（Githubのようなイメージですね。ちなみに、Webサーバーである Apacheのイメージ、データベースソフトである MySQLのイメージなんかもフリーで DockerHubにアップされています。）
コンテナイメージとは、端的には「アプリを動かすためのアプリ」と理解すればOKです。
プログラムコードや設定ファイルなどがひとまとめになっています。

そして、そのイメージをもとに**コンテナを実行**します。
Apache、MySQLのソフトウェアが実際に実行し、リクエストを受け付けている状態です。

管理対象のコンテナが少ない場合は **管理に手間がかかりません** 。
しかし、通常のシステムでは、コンテナは 1つではなく、負荷分散などを考慮して複数並行して動作させることが一般的です。

では次に、コンテナが複数起動している場合のスライドに移ります。

コンテナ管理

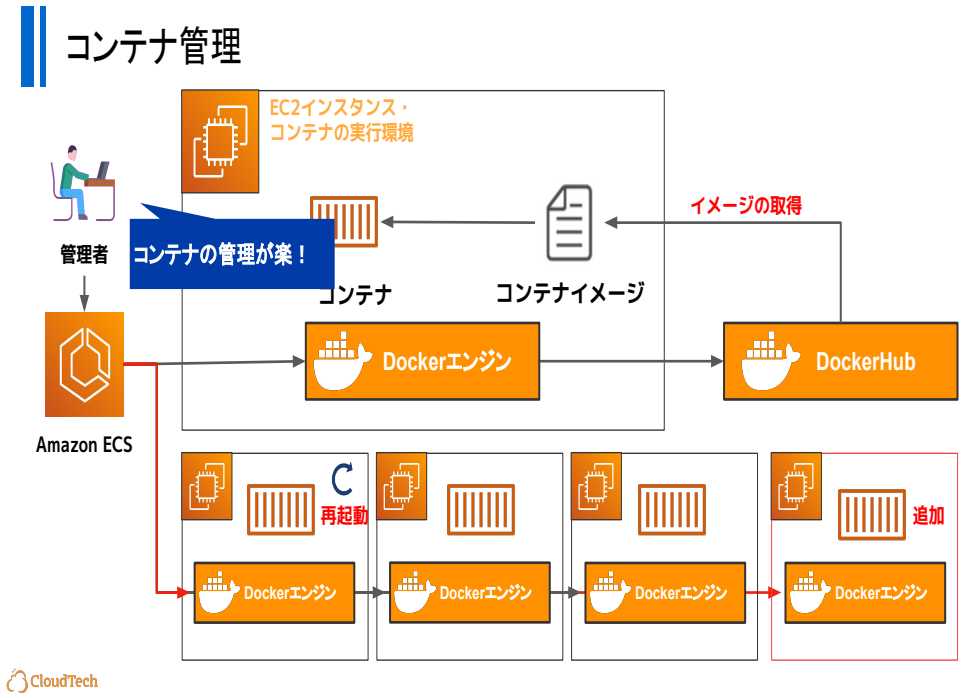


先ほどと違い、**複数の実行環境**があります。

管理者はコンテナを管理するために、**複数のDockerエンジンに命令を実行し、**コンテナを管理する必要が出てきます。
もしもコンテナが停止した場合には **再起動**したり、負荷が高まった時には**コンテナを追加で立ち上げる**といった運用も必要なります。

このように管理対象のコンテナが増えれば増えるほど、 **管理が辛く**なってしまいます。

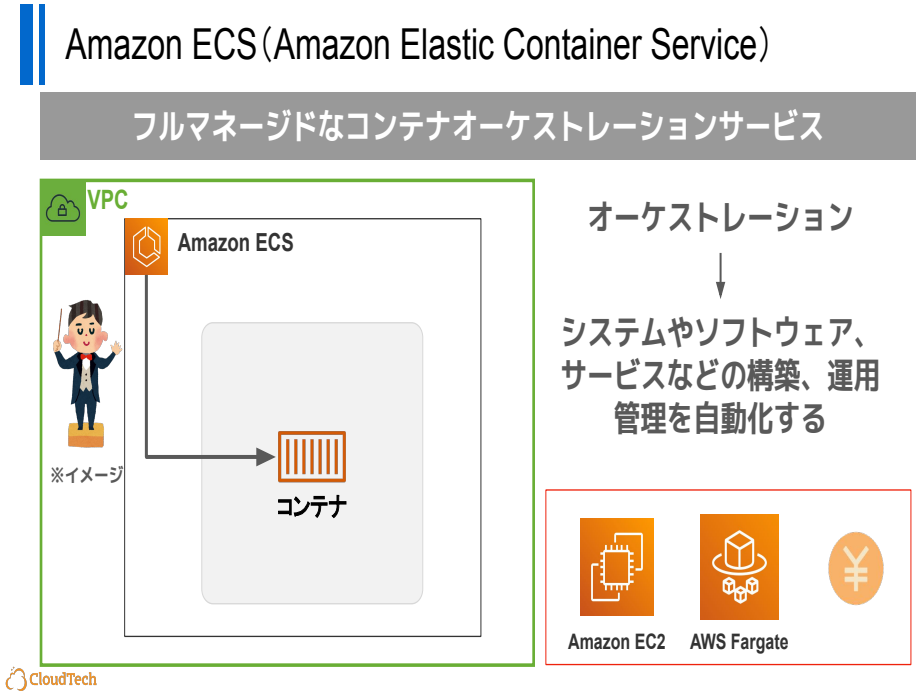
この課題にどう対応すれば良いのでしょうか？



ここで利用するサービスがコンテナ管理サービスの Amazon ECSです！
早速導入してみましょう！

ECSが複数のDockerエンジンに指示を出し、全てのコンテナを管理してくれます。
コンテナの**再起動**や**スケーリング**もあらかじめ定義した設定に従って、サービス提供可能な状態にしてくれます！

よって、管理者は煩雑だった管理作業から解放され、**ECSだけを管理**すれば良くなります。
つまり、ECSはコンテナを管理する上でに必要不可欠なサービスとなります。



改めてECSについて解説します。

ECSはAmazon Elastic Container Serviceと呼ばれ、コンテナ化されたアプリケーションのデプロイ、管理、スケーリングをしてくれる**フルマネージドなコンテナオーケストレーションサービス** です 。
フルマネージドということでサーバー管理や、ソフトウェアの更新などの運用に関わる設定を **ECSが代わり**に行ってくれます。
もしECSが無かったら手動でコンテナを管理しなければならないので、これは嬉しいですよね。

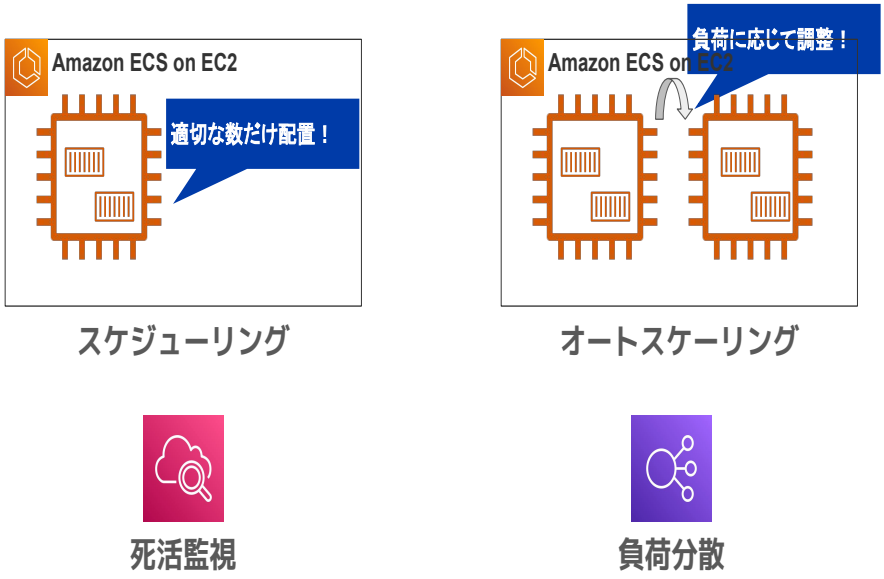
ここで普段聞き慣れない「 **オーケストレーション** 」という言葉が登場しました。
「オーケストレーション」とは「 **システムやソフトウェア、サービスなどの構築、運用管理を自動化する** 」ことを指しています。
「良い感じに管理してくれているんだな」と理解してもらえれば OK です。
オーケストラの指揮者 が、各楽器の演奏者に指揮を出している

様子を例えるとわかりやすいですね。

また、料金ですが、基本的なコンテナオーケストレーション機能は追加費用なしで提供されます。
ただし、使用したコンピューティングリソースである **EC2インスタンスや AWS Fargateの使用量に基づいて費用が発生します** 。
Fargateについては後続の講座で説明します。

では次にECSがどのような役割を果たすのかみていきましょう！

Amazon ECSの役割



コンテナオーケストレーションの役割を 4点お伝えします。

1つ目は「**スケジューリング**」です。
利用可能なリソースを考慮し、コンテナを自動で適切なサーバーに配置します。
これにより、アプリケーションが最適なパフォーマンスと効率で実行されるようになります。

2つ目は「**オートスケーリング**」です。
設定されたメトリクスに基づいて、コンテナインスタンス（つまりコンテナを実行する基盤である EC2インスタンス）の数を自動で調整します。
これにより、需要の増減に応じてリソースを効率的にスケールイン・アウトできます。

3つ目は「**死活監視**」です。
コンテナの状態を監視し、問題が発生した場合には自動的にコ

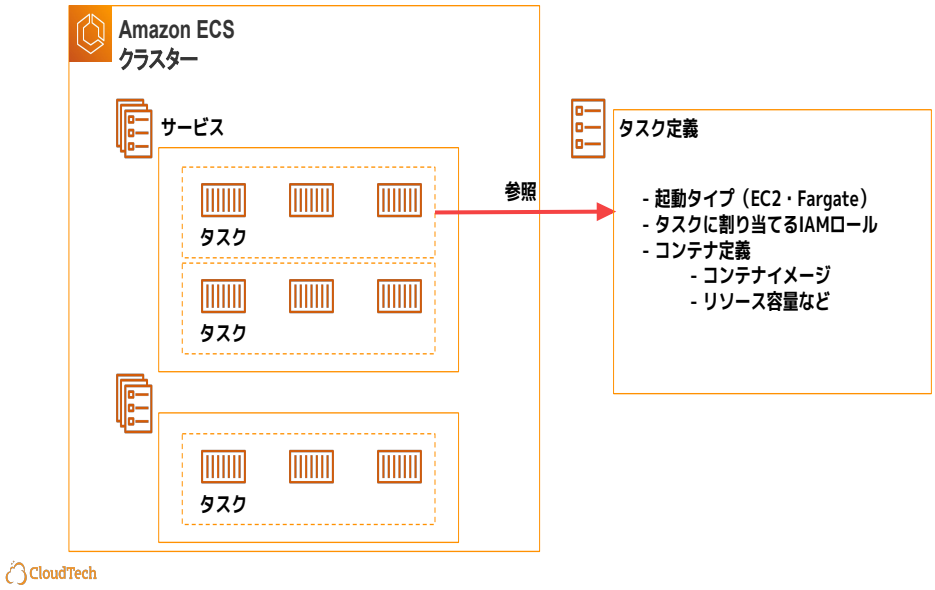
ンテナを再起動または置換えます。
これにより、アプリケーションの可用性と信頼性を向上させることができます。

4つ目は「**負荷分散（ロードバランシング）**」です。
ECSは、Amazon Elastic Load Balancing (ELB) と統合でき、複数のコンテナインスタンス間で負荷を均等に分けることができます。
これにより、アプリケーションが安定して高速に応答することが可能になります。

このようにECSを使用することで、開発者はアプリケーションのデプロイメント、管理、スケーリングを簡単に行い、システムの高可用性と高性能を実現できます。

ここまでで、なぜECSが必要なのか？ ECSの役割は理解できましたでしょうか？
それでは次に、ECSの構成要素をみてみましょう！

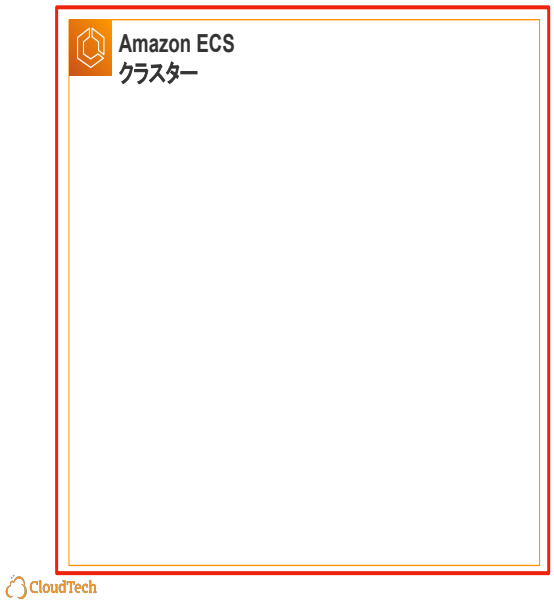
Amazon ECSの構成要素



ECSは様々な構成要素で成り立っています。
大きく4つ構成要素があります。

クラスター・タスク定義・サービス・タスクです。
ここでは各要素について順番に解説していきます。
またイメージしやすいように、少し大胆な切り口ですが、各要素をレストランに例えながら解説していきます！

Amazon ECSの構成要素



※イメージ

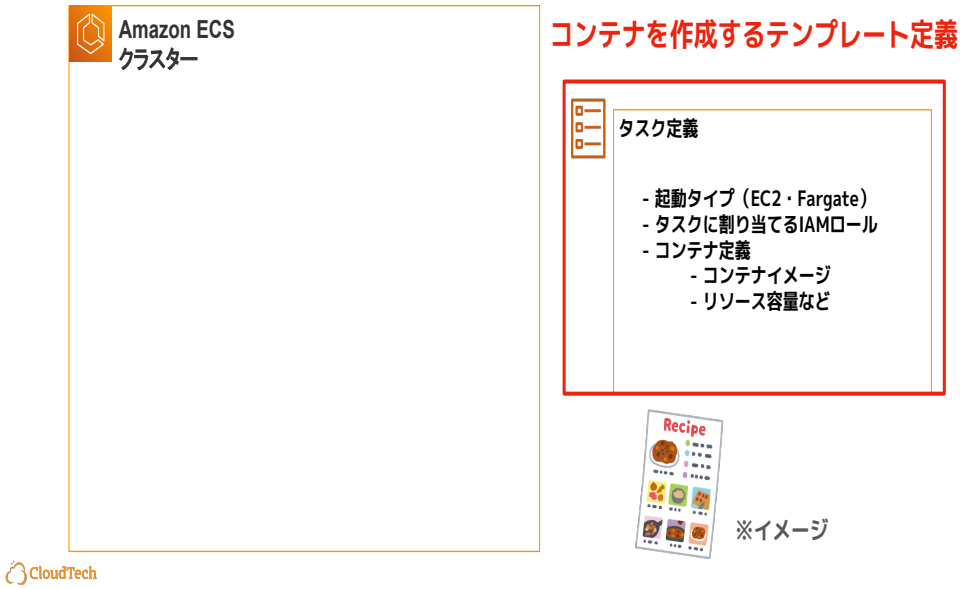
まず**クラスター**です。
クラスターはECSで**コンテナを動かすための実行環境** です。

クラスターを作成することで、システムの大まかな区分けや IAM 権限の境界を設定することができます。
クラスターを複数作成することで、システムごとに ECSの設定や権限管理などを分けることができます。

実行環境となるので、**レストラン**をイメージしてみましょう。

世の中にはいろいろなアプリがあります。通販やマッチングサイト、SNS…
アプリの提供価値を「料理」とするのであれば、複数のレストランごとに店舗を分ける必要がありますよね。
クラスターも同様に、コンテナの実行環境を分けることでセキュリティ境界を分けることができます。

Amazon ECSの構成要素

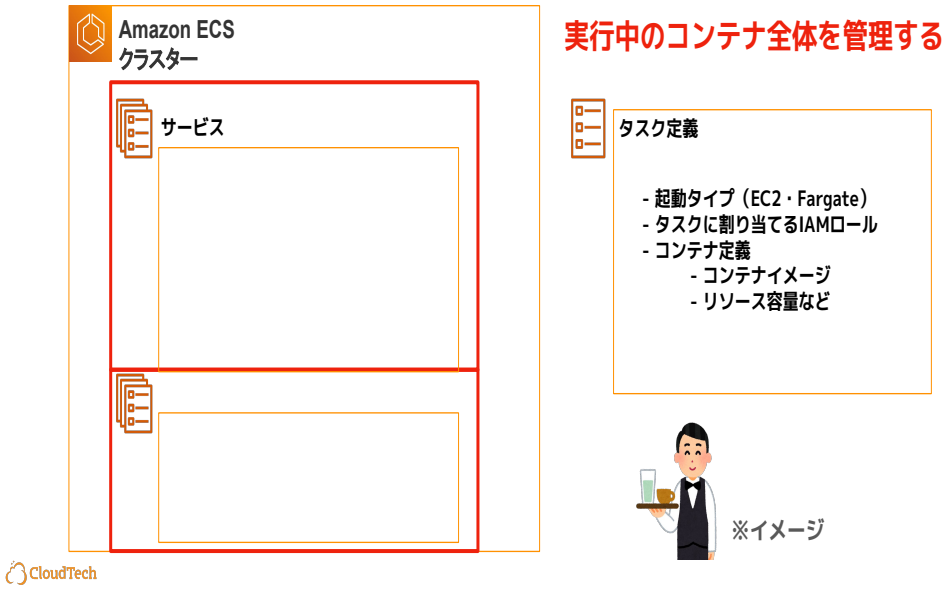


次に**タスク定義**です。
タスク定義は**コンテナを作成する定義、つまりテンプレートとなるテキスト情報をイメージしてください**。

コンテナを実行するために必要な設定や要件を、設定ファイルで定義しています。
具体的には、起動タイプ（EC2・Fargate）・タスクに割り当てるIAMロール・コンテナ定義（コンテナイメージ（Webサーバーであるapacheのイメージ、MySQLのイメージなどフリーでネットに配布されています）・CPUやメモリなどのリソース容量など）を定義します。
Docker講座を受講済の方は、Dockerfileと思っていただければOKです。

テンプレートとなるので、**料理のレシピ**をイメージしてみましょう。
タスク定義はコンテナを実行するために必要な情報を定義してます。

Amazon ECSの構成要素

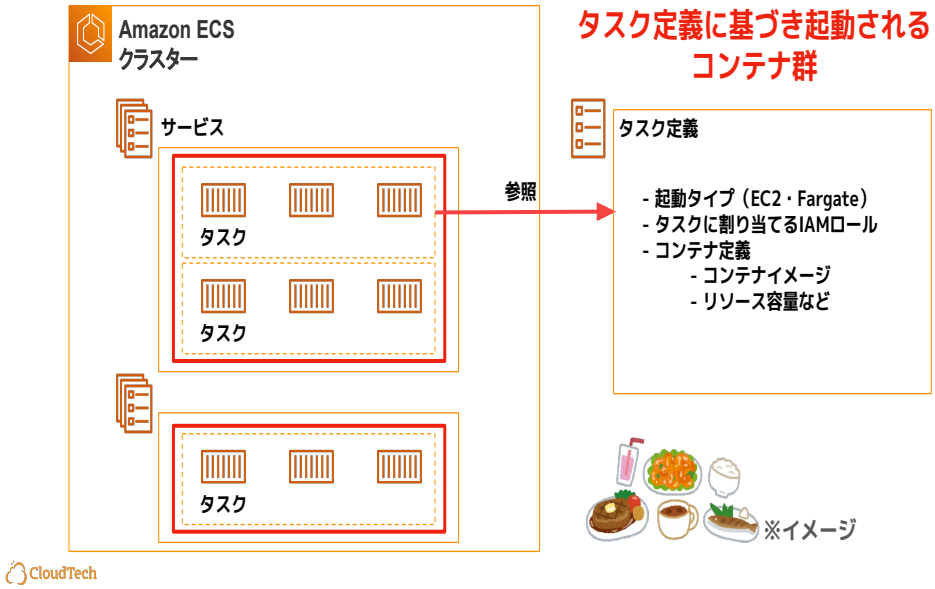


次に**サービス**です。
サービスは**実行中のコンテナ全体を管理する機能** を持っています。

実行するコンテナ数を定義したり、どのロードバランサーと連携するかを設定したり、ネットワーク構成を指定したりと、様々な管理を行ってくれます。

全体管理となるので、料理を提供する **ウェイター**をイメージしてみましょう。
ウェイターは客の注文に応じて料理を提供し、料理がなくなったら厨房（クラスター）から新しい料理（コンテナ）を取りに行きます。
サービスも同じく、コンテナを継続的に実行し、必要に応じて新しいコンテナを起動します。

Amazon ECSの構成要素



最後に**タスク**です。
タスクは**タスク定義に基づき起動されるコンテナ群** をさしています。

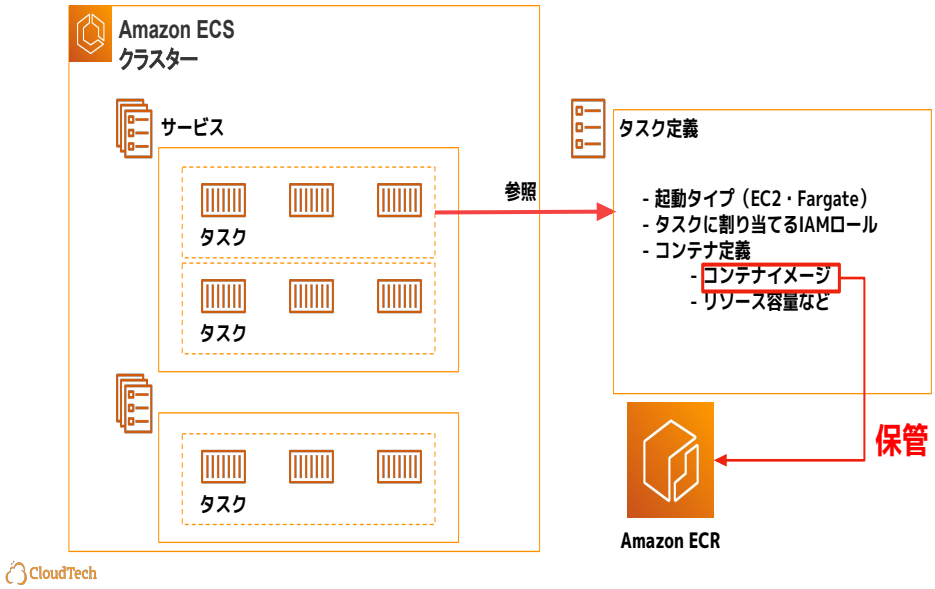
アプリケーションの実行単位を指し、1つのタスクは1つ以上のコンテナから構成されていることがポイントです。
例えば、Webサーバーである Apacheのタスク、nginxのタスク、MySQLのタスクなどがあります。

アプリケーションなので、**調理された料理** そのものをイメージしてみましょう。
レシピに従って材料を組み合わせ、調理した結果が料理です。
同様に、タスク定義に基づいて実際に実行されるコンテナ群がタスクとなります。

以上がECSの主な構成要素となります。
それでは、これらの構成要素がどのように組み合わさってコンテ

ナが起動しているかみてみましょう！

Amazon ECSの構成要素



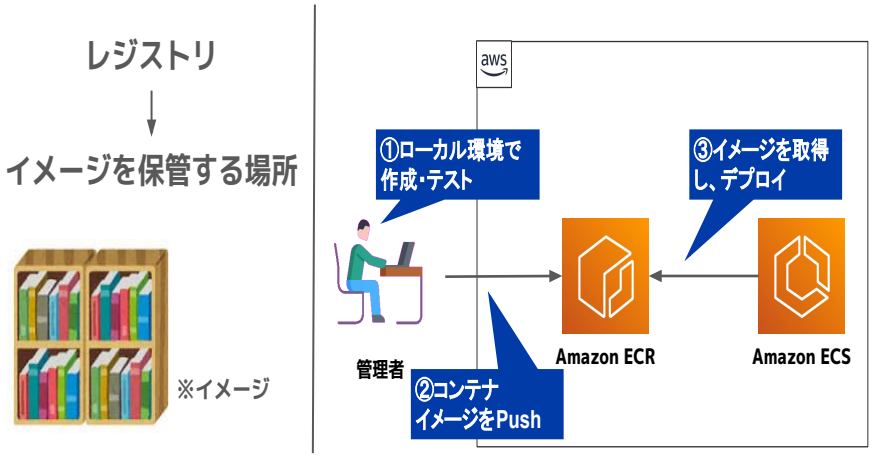
基本的にはこのような流れでコンテナを起動します。

1.まずはECSを実行する基盤となる**ECSクラスターを作成**します。
2.次にECSタスクを起動するテンプレートとなる **タスク定義**を作成します。
3.次に実行中のタスクを管理する **ECSサービス**を作成します。
4.するとサービスで定義したタスク数に基づいて **ECSタスク**が自動で起動されます。

ここで1点疑問が生じます。
タスク定義に書かれている**コンテナイメージ**はどこから取得してくるのでしょう？
ECSでは、通常Dockerイメージの保管に利用されるサービスである**Amazon ECR**から取得します。

Amazon ECR（Amazon Elastic Container Registry）

コンテナイメージを格納するフルマネージド型のレジストリサービス



CloudTech

Amazon ECRとはAmazon Elastic Containe Registry（ECR）と呼ばれ、
AWSが提供している**コンテナイメージを格納するフルマネージド型のコンテナレジストリサービス** です。

ここでも普段聞き慣れない「**レジストリ**」という言葉が登場しましたね。
「レジストリ」とは**イメージを保管する場所** のことを指します。イメージとしては**本棚**を想像してもらえればOKです。

では、**ECRの使い方**を簡単にみてみましょう。

1. まずユーザーはコンテナイメージを **ローカル環境で作成・テスト**します。
例えば、フリーのDockerHubからダウンロードしたnginxのコンテナイメージをローカルの環境で動かし、設定ファイルをいじくり動かしてみてテスト完了、そして再びコンテナイメージとしてビルド

（圧縮）するといった流れです。
2. 次にそのコンテナイメージを **ECRにpushします。**pushとはアップロードすることです。
3. そして、ECSでコンテナが起動される時にECRから**コンテナイメージを取得してコンテナをデプロイ**します。

このようにECRを使用することで、開発者はコンテナイメージを AWS上の安全な場所に保管し、必要に応じて簡単にデプロイに利用することができます。
これにより、コンテナ化されたアプリケーションの開発とデプロイの効率が大幅に向上します。

最後にAmazon ECSのまとめとなります。ECSは

- AWSが提供するフルマネージド型の **コンテナオーケストレーションサービス**
- **Amazon ECSの役割**
  - スケジューリング
  - オートスケーリング
  - 死活監視
  - 負荷分散
- ECSには**クラスター・タスク定義・ タスク・サービス**といった構成要素が含まれる。

ECSを利用することで、コンテナの管理から解放され、アプリケーションの開発・構築に集中することが出来るようになります。コンテナを管理する上で必要不可欠なサービスであるため、ハンズオンも積極的にやっていきましょう！

【公式ドキュメント】
https://docs.aws.amazon.com/ja_jp/AmazonECS/latest/developerguide/Welcome.html