

ITA\_利用手順マニュアル

Terraform-CLI-driver

*－*第1.11版*－*

Copyright © NEC Corporation 2023. All rights reserved.

免責事項

本書の内容はすべて日本電気株式会社が所有する著作権に保護されています。

本書の内容の一部または全部を無断で転載および複写することは禁止されています。

本書の内容は将来予告なしに変更することがあります。

日本電気株式会社は、本書の技術的もしくは編集上の間違い、欠落について、一切責任を負いません。

日本電気株式会社は、本書の内容に関し、その正確性、有用性、確実性その他いかなる保証もいたしません。

商標

* LinuxはLinus Torvalds氏の米国およびその他の国における登録商標または商標です。
* Red Hatは、Red Hat, Inc.の米国およびその他の国における登録商標または商標です。
* Apache、Apache Tomcat、Tomcatは、Apache Software Foundationの登録商標または商標です。
* Terraformは、HashiCorpの登録商標または商標です。

その他、本書に記載のシステム名、会社名、製品名は、各社の登録商標もしくは商標です。

なお、® マーク、TMマークは本書に明記しておりません。

※本書では「Exastro IT Automation」を「ITA」として記載します。

目次

[目次 2](#_Toc124263693)

[はじめに 3](#_Toc124263694)

[1 Terraform-CLI-driver概要 4](#_Toc124263695)

[1.1 Terraformについて 4](#_Toc124263696)

[1.2 Terrform-CLI-driverについて 4](#_Toc124263697)

[2 Terraform-CLI-driverでの変数取り扱い 5](#_Toc124263698)

[2.1 変数の種類 5](#_Toc124263699)

[2.2 変数の抜出および具体値登録 5](#_Toc124263700)

[2.3 変数のタイプについて 6](#_Toc124263701)

[3 Terraform-CLI-driver コンソールメニュー構成 11](#_Toc124263702)

[3.1 メニュー/画面一覧 11](#_Toc124263703)

[4 Terraform-CLI-driver利用手順 13](#_Toc124263704)

[4.1 Terraform作業フロー 13](#_Toc124263705)

[5 Terraform-CLI-driver機能・操作方法説明 15](#_Toc124263706)

[5.1 基本コンソール 15](#_Toc124263707)

[5.1.1 オペレーション一覧 15](#_Toc124263708)

[5.2 Terraform-CLI-driverコンソール 16](#_Toc124263709)

[5.2.1 インターフェース情報 16](#_Toc124263710)

[5.2.2 Workspaces管理 18](#_Toc124263711)

[5.2.3 Movement一覧 20](#_Toc124263712)

[5.2.4 Module素材集 22](#_Toc124263713)

[5.2.5 Movement-Module紐付 24](#_Toc124263714)

[5.2.6 変数ネスト管理 26](#_Toc124263715)

[5.2.7 代入値自動登録設定 28](#_Toc124263716)

[5.2.8 代入値管理 31](#_Toc124263717)

[5.2.9 作業実行 33](#_Toc124263718)

[5.2.10 作業状態確認 35](#_Toc124263719)

[5.2.11 作業管理 38](#_Toc124263720)

[6 構築コード記述方法 39](#_Toc124263721)

[6.1 Moduleの記述 39](#_Toc124263722)

[6.2 BackYard コンテンツ 39](#_Toc124263723)

[7 運用操作 41](#_Toc124263724)

[7.1 メンテナンス 41](#_Toc124263725)

[7.2 メンテナンス方法について 42](#_Toc124263726)

[8 付録 43](#_Toc124263727)

[8.1 Module素材記入例・登録例 43](#_Toc124263728)

[8.2 変数ネスト管理フロー例 57](#_Toc124263729)

はじめに

本書では、ITAの機能および操作方法について説明します。

# Terraform-CLI-driver概要

本章ではTerraformおよびTerraform-CLI-driverについて説明します。

## Terraformについて

TerraformとはHashiCorp社が提供するインフラストラクチャを効率化するオーケストレーションツールです。

HCL(HashiCorp Configuration Language)という言語でコード化したインフラストラクチャ構成について、実行計画を生成したうえで構築を実行します。

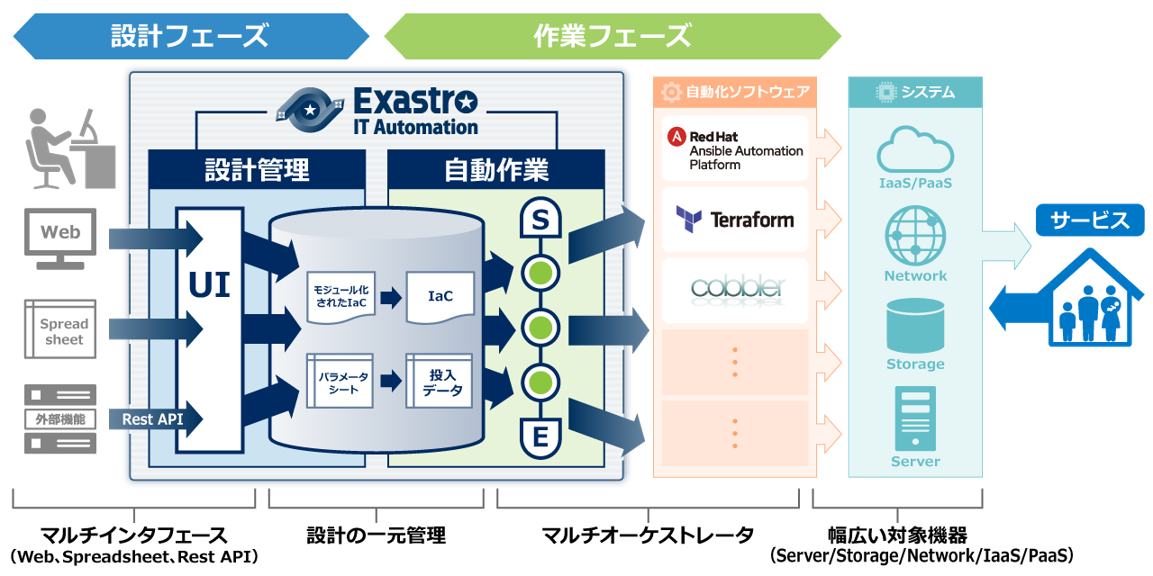
Terraform-CLI-driverでは、HashiCorp社が提供するTerraform CLIパッケージをITAと同一サーバにインストールし、利用します。

Terraformの詳細情報については、Terraformの製品マニュアルを参照してください。

## Terrform-CLI-driverについて

Terraform-CLI-deriverはITAシステムのオプションとして機能し、ITAと同一のサーバにインストールしたTerraformに対し、作業の実行（Plan / Apply）および作業ログの取得を行うことができます。

作業の実行（Plan / Apply）に利用するModuleファイルをITAシステム上で部品化し、再利用できるよう管理することができます。



**図 1.2-1 ITAシステム概要**

また、Terraform-CLI-driverは、Module中の変数を画面から設定することができます。詳細は本書「2 Terraform-CLI-driverでの変数取り扱い」をご参照ください。

# Terraform-CLI-driverでの変数取り扱い

## 変数の種類

Terraform-CLI-driverでは、Module中の変数の具体値をITAの設定画面から設定することができます。

**※設定方法の詳細は、本書 「5.2.8 代入値管理」を参照してください。**

Module中の変数で、ITAの変数として扱える変数は以下の1種類があります。

|  |  |
| --- | --- |
| 種類 | 内容 |
| 通常変数 | 変数名に対して具体値を定義できる変数です。  Module内の変数は HCL（HashiCorp Configuration Language）の変数ルールに従い 以下の形式で記述してください。この場合「xxx」がModuleから変数として抜出されます。  また、typeとdefault値を設定することができます。  この場合、「○○」がtypeとして、「△△」がdefaultとして抜出されます。  typeとdefaultの設定は必須ではありません。   |  | | --- | | variable “xxx” {  type = ○○  default = △△  ～省略~  } | |

## 変数の抜出および具体値登録

ITAにアップロードされたModule素材から変数を抜出して具体値を登録できます。

抜出した変数の具体値は「5.2.7代入値自動登録設定」「5.2.8代入値管理」で具体値を登録します。

登録された変数と具体値は、作業実行時に生成されるterraform.tfvarsファイルに「変数名」が「Key」、「具体値」が「Value」として記載され、作業実行で使用されます。

## 変数のタイプについて

変数内でtypeを設定することができます。

Module内の変数は HCL（HashiCorp Configuration Language）の変数ルールに従い記述してください。

ITA内で扱う変数は以下の通りです。

記入例は「 8.1 Module素材記入例」をご参照ください。

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **type** | **詳細** | **代入順序**  **対象※1** | **メンバー変数**  **対象※2** | **typeの**  **記述例** | **defaultの記述例** |
| string | 文字列型。 | × | × | string | あいう |
| number | 数値型。 | × | × | number | 2022 |
| bool | trueまたはfalse | × | × | bool | true |
| list | 配列型。 | 〇 | × | list(string) | [“あ”, “い”, “う”] |
| set | 配列型。ユニークな値の設定が求められる。  ITA上では具体値がユニークか否かは判定されません。 | 〇 | × | set(number) | [1, 2, 3] |
| tuple | 配列型。予めn番目にどのtypeを設定するか決めておく必要があります。  値の入力数が決められているのでITA上ではメンバー変数としてプルダウンで選択します。 | × | 〇 | tuple([string, number]) | [“あいう”, 2022] |
| map | key-value型。ITA上ではmap型が一つ以上含まれているtypeを設定した場合、type情報からKEY値を特定できないので、代入値を設定する場合はHCL設定をONにしてください。  HCL設定についての説明は「5.2.7代入値自動登録設定」または「5.2.8代入値管理」をご参照ください。 | × | × | map(string) | { “key” = “value” } |
| object | key-value型。  ITA上ではkeyをメンバー変数として扱います。key名に日本語は含まないでください。 | × | 〇 | object({  key = number  }) | {  “key” = 2022  } |
| any | すべてに適合する型。  ITA上ではstring型と同じ扱いとなります。 | × | × | any | あいう |
| 記載なし | typeを記載しなかった場合、ITA上ではstring型と同じ扱いになります。 | × | × |  | あいう |

※１…代入順序

変数に複数具体値を設定する際の先頭から代入する順序。

変数または階層構造の変数の最下層の変数のタイプがlist,setの場合、代入値自動登録設定メニュー/代入値管理メニューにて設定可能。

例：変数のタイプがlistの場合

・tfファイルと登録値

|  |
| --- |
| variable "VAR\_hoge" {  type = list(string)  } |

1. 代入値例（代入値自動登録設定/代入値管理）

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **項番** | **変数名** | **メンバー変数** | **代入順序** | **具体値** |
| 1 | VAR\_hoge | 入力不要 | 1 | あいう |
| 2 | VAR\_hoge | 入力不要 | 2 | かきく |

２．Terraformに送信される値

|  |
| --- |
| [“あいう”, “かきく”] |

例：階層構造の変数の最下層の変数のタイプがsetの場合

・tfファイルと登録値

|  |
| --- |
| variable "VAR\_hoge" {  type = object({  key = set(number)  })  } |

1. 代入値例（代入値自動登録設定/代入値管理）

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **項番** | **変数名** | **メンバー変数** | **代入順序** | **具体値** |
| 1 | VAR\_hoge | key | 1 | 1 |
| 2 | VAR\_hoge | key | 2 | 2 |

２．Terraformに送信される値

|  |
| --- |
| {  key = [1, 2]  } |

※２…メンバー変数

変数がkey-value型である場合のkey名。変数のタイプがobjectの場合、<KEY> = <TYPE>の<KEY>をメンバー変数とする。

変数のタイプがtupleの場合、tuple内に定義した変数を先頭から[0],[1],[2]…と採番してメンバー変数とする。

変数のタイプが変数ネスト管理メニューの登録対象の場合、最大繰返数をもとに[0],[1],[2]…と採番してメンバー変数とする。

変数ネストに関しては「5.2.6変数ネスト管理」をご参照ください。

例：変数のタイプがobjectの場合

・tfファイルと登録値

|  |
| --- |
| variable "VAR\_hoge" {  type = object({  NAME = string,  IP = string  })  default = {  “NAME” = “machine\_01”,  “IP” = “127.0.0.1”  }  } |

1. 代入値例（代入値自動登録設定/代入値管理）

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **項番** | **変数名** | **メンバー変数** | **代入順序** | **具体値** |
| 1 | VAR\_hoge | NAME | 入力不可 | my\_machine |
| 2 | VAR\_hoge | IP | 入力不可 | 192.168.0.1 |

２．Terraformに送信される値

|  |
| --- |
| {  NAME = “my\_machine”,  IP = “192.168.0.1”  } |

例：変数のタイプが変数ネスト管理対象の場合

・tfファイルと登録値

|  |
| --- |
| variable "VAR\_hoge" {  type = list(set(string))  default = [  [“aaa”, “bbb”],  [“ccc”, “ddd”]  ]  } |

1. 代入値例（代入値自動登録設定/代入値管理）

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **項番** | **変数名** | **メンバー変数** | **代入順序** | **具体値** |
| 1 | VAR\_hoge | [0] | 1 | あああ |
| 2 | VAR\_hoge | [0] | 2 | いいい |
| 3 | VAR\_hoge | [1] | 1 | ううう |
| 4 | VAR\_hoge | [1] | 2 | えええ |

２．Terraformに送信される値

|  |
| --- |
| [  [“あああ”, “いいい”],  [ “ううう”, “えええ”]  ] |

例：変数のタイプがtupleの場合

・tfファイルと登録値

|  |
| --- |
| variable "VAR\_hoge" {  type = tuple([string, number])  default = [“aaa”, 2022]  } |

1. 代入値例（代入値自動登録設定/代入値管理）

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **項番** | **変数名** | **メンバー変数** | **代入順序** | **具体値** |
| 1 | VAR\_hoge | [0] | 入力不可 | bbb |
| 2 | VAR\_hoge | [1] | 入力不可 | 2023 |

２．Terraformに送信される値

|  |
| --- |
| [“bbb”, 2023: |

# Terraform-CLI-driver コンソールメニュー構成

本章では、ITAコンソールのメニュー構成について説明します

## メニュー/画面一覧

1. ITA**基本コンソールのメニュー**

Terraform-CLI-driverで利用するITA基本コンソールのメニュー一覧を以下に記述します。

表 3.1‑1基本コンソール メニュー/画面一覧

|  |  |  |  |
| --- | --- | --- | --- |
| No | メニューグループ | メニュー・画面 | 説明 |
| 1 | ITA 基本コンソール | 投入オペレーション  一覧 | オペレーション一覧をメンテナンス(閲覧/登録/更新/廃止)できます。 |

1. **Terraform-CLI-driverコンソールのメニュー**

Terraform-CLI-driverコンソールのメニュー一覧を以下に記述します。

表 3.1‑2Terraform-CLI-driverコンソール メニュー/画面一覧

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| No | メニューグループ | メニュー・画面 | 非表示メニュー※１ | 説明 |
| 1 | Terraform | インターフェース情報 |  | 作業実行の情報を管理します。 |
| 2 | Workspaces管理 |  | Terraformで利用するWorkspaceの情報を管理します。 |
| 3 | Movement一覧 |  | Symphonyに登録するMovementの一覧を管理します。 |
| 4 | Module素材集 |  | Moduleファイルを管理します。 |
| 5 | Movement-Module紐付 |  | MovementとModule素材の関連付けを管理します。 |
| 6 | 変数ネスト管理 |  | Module素材集で登録したtfファイルで定義されている変数のタイプがlist,setかつ、その変数の中でlist,set,tuple,objectが定義されている場合、メンバー変数の最大繰返数を管理します。 |
| 7 | 代入値自動登録設定 |  | パラメータシートのメニューに登録されているオペレーション毎の項目や値を紐付けるMovement と変数を管理します |
| 8 | 代入値管理 |  | 変数の代入値を管理します。 |
| 9 | 作業実行 |  | 作業実行するMovementとオペレーションを選択し実行を指示します。 |
| 10 | 作業状態確認 |  | 作業実行状態を表示します。 |
| 11 | 作業管理 |  | 作業実行履歴を管理します。 |
| 12 | Module変数紐付管理 | 〇 | Module変数とModule素材の紐付を管理します。 |
| 13 | メンバー変数管理 | 〇 | メンバー変数を管理します。 |
| 14 | Movement変数紐付管理 | 〇 | Movementと変数名の紐付を管理します。 |

※1非表示メニューは、バックヤード機能でデータの登録・更新を行うメニューです。

Terraform-CLI-driver機能をインストールした状態では表示されないメニューに設定されています。

非表示メニューを表示するには、「管理コンソール/ロール・メニュー紐付管理」で各メニューの復活処理を行います。詳細は「利用手順マニュアル\_管理コンソール」 を参照してください。

尚、データの更新を行うとバックヤード機能が正しく動作しなくなります。データの更新はしないで下さい。

# Terraform-CLI-driver利用手順

各Terraformコンソールの利用手順について説明します

## Terraform作業フロー

各Terraformコンソールにおける標準的な作業フローは以下のとおりです。

各作業の詳細は次項に記載しています。

ITA基本コンソールの利用方法は、「利用手順マニュアル\_基本コンソール」を参照してください。

以下は、Terraformで作業を実行するまでの流れです。

1. **投入オペレーション名の登録**

**④ 作業パターン(Movement)の登録**

**⑤ Module素材の登録**

**必須タスク**

**任意タスク**

**【凡例】**

**⑥　MovementにModule素材を指定**

**③ Workspaceの登録と連携**

**⑦ 最大繰返数の設定**

**⑧ 変数値の設定**

**② インターフェース情報の設定**

**⑨ 作業実行**

**⑩ 実行状態確認**

**⑪ 作業履歴確認**

* **作業フロー詳細と参照先**

1. **投入オペレーション名の登録**

ITA基本コンソールのオペレーション一覧の画面から、作業用の投入オペレーション名を登録します。

詳細は「5.1.1オペレーション一覧」を参照してください。

1. **インターフェース情報の設定**

作業実行の情報を設定します。

詳細は「5.2.1インターフェース情報」を参照してください。

1. **Workspaceの登録と連携**

Workspaceの情報を登録します。

詳細は「5.2.2 Workspaces管理」を参照してください。

1. **作業パターン(Movement)の登録**

作業用のMovementを登録します。  
詳細は「5.2.3 Movement一覧」を参照してください。

1. **Module素材の登録**作業で実行するModuleファイルを登録します。  
   詳細は「5.2.4 Module素材集」を参照してください。
2. **MovementにModule素材を指定**

登録したMovementにModule素材を指定します。

詳細は「5.2.5 Movement-Module紐付」を参照してください。

1. **最大繰返数の設定（必要に応じて実施）**

メンバー変数の最大繰返数を設定します。

詳細は「5.2.6変数ネスト管理」を参照してください。

1. **変数値の設定（必要に応じて実施）**

Movementに登録したModule素材内で定義した変数の値を設定します。変数を利用していない場合、設定は不要です。

詳細は「5.2.7代入値自動登録設定」「5.2.8代入値管理」を参照してください。

1. **作業実行**

実行日時、投入オペレーションを選択して設定して処理の実行を指示します。

詳細は「5.2.9作業実行」を参照してください。

1. **作業状態確認**実行した作業の状態がリアルタイムで表示されます。

また、作業の緊急停止や、実行ログ、エラーログを監視することができます。  
詳細は「5.2.10作業状態確認」を参照してください。

1. **作業履歴確認**実行した作業の一覧が表示され履歴が確認できます。

詳細は「5.2.11作業管理」を参照してください。

# Terraform-CLI-driver機能・操作方法説明

本章では、Terraform-CLI-driverで利用する各コンソールの機能について説明します。

## 基本コンソール

本節では、ITA基本コンソールでの操作について記載します。

本作業はITA基本コンソールマニュアルを参照して、ITA基本コンソール画面内で作業を実施してください。

### オペレーション一覧

1. [オペレーション一覧]画面では、オーケストレータで実行する作業対象ホストに対するオペレーションを管理します。作業はITA基本コンソール内メニューより選択します。



図 5.1.1- 1 サブメニュー画面（オペレーション一覧）

登録方法の詳細は、関連マニュアルの「利用手順マニュアル\_基本コンソール」をご参照下さい。

## Terraform-CLI-driverコンソール

本節では、Terraform-CLIコンソールでの操作について記載します。

### インターフェース情報

1. [インターフェース情報]では、ITAシステムと連携するTerraformの作業実行時の情報をメンテナンス（閲覧／更新）することができます。

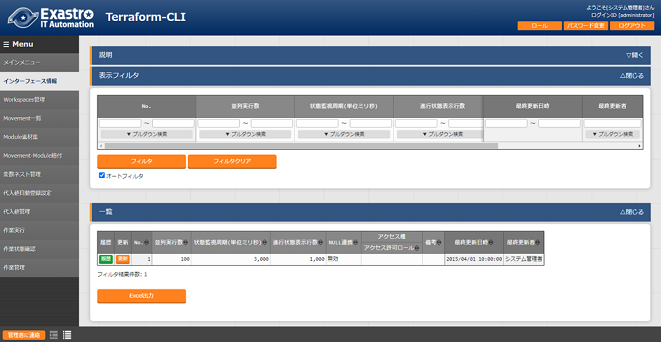


図 5.2.1- 1 サブメニュー画面（インターフェース情報）

1. 「一覧」-「更新」ボタンより、インターフェース情報の登録を行います。

インターフェース情報が未登録または、複数レコード登録されている状態で作業実行した場合、**作業実行は想定外エラーとなります**。



図 5.2.1- 2登録画面（インターフェース情報）

1. インターフェース情報画面の項目一覧は以下のとおりです。

**表 5.2.1‑1　項目一覧（インタフェース情報）**

| **項目** | **説明** | **入力**  **必須** | **入力形式** | **制約事項** |
| --- | --- | --- | --- | --- |
| 並列実行数 | 同時に実行できるMovement(Teraform-CLI)の最大数を入力します。 | ○ | 手動入力 |  |
| 状態監視周期（単位ミリ秒） | 「5.2.10作業状態確認」で表示されるログのリフレッシュ間隔を入力します。通常は3000ミリ秒程度が推奨値です。 | ○ | 手動入力 | 最小値 1000 ミリ秒 |
| 進行状態表示行数 | 「5.2.10作業状態確認」での進行ログ・エラーログの最大表示行数を入力します。  ステータスが[未実行]、[準備中]、[実行中]、[実行中(遅延)]の場合、指定した行数でログを出力します。  ステータスが[完了]、[完了(異常)]、[想定外エラー]、[緊急停止]、[未実行(予約)]、[予約取消]の場合、指定した行数ではなくすべてのログを出力します。  環境毎にチューニングを要しますが、通常は1000行程度が推奨値です。 | ○ | 手動入力 | - |
| NULL連携 | 代入値自動登録設定でパラメータシートの具体値が NULL(空白)の場合に 、代入値管理への登録 を NULL(空白)の値で行うか設定します。代入値自動登録設定メニューの「NULL 連携」が空白の場合この値が適用されます。  ・「有効」の場合、パラメータシートの値がどのような 値でも代入値管理への登録が行われます。  ・「無効」の場合、パラメータシートに値が入っている 場合のみ代入値管理への登録が行われます。 | ○ | リスト選択 |  |
| 備考 | 自由記述欄です。 | - | 手動入力 | 最大長4000バイト |

### Workspaces管理

1. [Workspaces管理]では、Terraformで利用するWorkspaceについてのメンテナンス（閲覧／登録／更新／廃止／リソース削除）を行います。

WorkspaceはTerraformコマンドを実行するためのディレクトリとして利用します。

同一のWorkspaceを対象とした作業実行を行う場合、Terraformが生成するstateファイルはWorkspace単位で管理され、冪等性が保たれます。

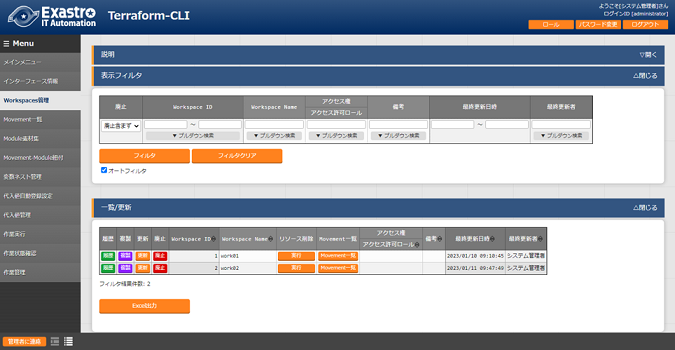


図 5.2.22‑1 サブメニュー画面（Workspaces管理）

1. 「登録」-「登録開始」ボタンより、Workspace情報の登録を行います。



図 5.2.22-2 登録画面（Workspaces管理）

1. リソース削除ボタンをクリックすると、対象のWorkspaceに対してリソース削除(terraform destroy)が実行されます。

Movement一覧ボタンをクリックすると、対象のWorkspaceの「5.2.3 Movement一覧」へ遷移します。



図 5.2.22‑3 Workspaces管理

1. Workspaces管理画面の項目一覧は以下のとおりです。

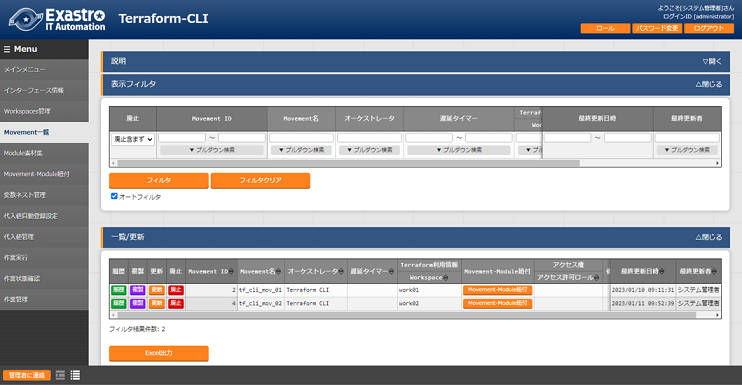
**表 5.2.22-1項目一覧（Workspaces管理）**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **項目** | **説明** | **入力**  **必須** | **入力形式** | **制約事項** |
| Workspace Name | Workspaceの名前を入力します。  半角英数字と記号 \_ - （アンダーバーとハイフン）のみ利用可能です。 | ○ | 手動入力 | 最大長90バイト |
| リソース削除 | Workspaceごとに構成・管理されたリソースの削除を実行するボタンです。  クリックすると確認ダイアログが表示され[OK]をクリックすると「5.2.10作業状態確認」に遷移し、対象のWorkspaceごとに構成・管理されたリソースの削除が実行されます。 | - | - |  |
| Movement一覧 | 「5.2.3 Movement一覧」へ遷移するボタンです。 | - | - |  |
| 備考 | 自由記述欄です。 | - | 手動入力 | 最大長4000バイト |

### Movement一覧

1. [Movement一覧]では、Movement名についてのメンテナンス（閲覧／登録／更新／廃止）を行います。

MovementはTerraform利用情報としてWorkspaceと紐付ける必要があるため、先に「5.2.2 Workspaces管理」で対象を登録しておく必要があります。



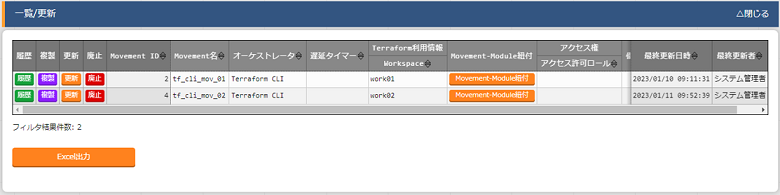
**図5.2.33-1サブメニュー画面（Movement一覧）**

1. 「登録」-「登録開始」ボタンより、Movement情報の登録を行います。



**図 5.2.33-2 登録画面（Movement一覧）**

1. Movement-Module紐付ボタンをクリックすると対象のMovementの「5.2.5 Movement-Module紐付」へ遷移します。



**図5.2.33-3サブメニュー画面（Movement一覧）**

1. Movement一覧画面の項目は以下の通りです。

**表 5.2.33-1 項目一覧（Movement一覧）**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **項目** | | **説明** | **入力**  **必須** | **入力形式** | **制約事項** |
| Movement名 | | Movementの名称を入力します。 | ○ | 手動入力 | 最大長256バイト |
| オーケストレータ | | 『Terraform CLI』が自動で入力されます。 | - | - | - |
| 遅延タイマー | | Movementが指定期間遅延した場合にステータスを遅延として警告表示したい場合に指定期間(1～)を入力します。(単位:分)  未入力の場合は警告表示しません。 | - | 手動入力 | - |
| Terraform利用情報 | Workspace | 「5.2.2 Workspaces管理」にて登録したWorkspaceを選択します。 | ○ | リスト選択 |  |
| Movement-Module紐付 | | 「5.2.5 Movement-Module紐付」へ遷移するボタンです。 | - | - |  |
| 備考 | | 自由記述欄です。 | - | 手動入力 | 最大長4000バイト |

### Module素材集

1. [Module素材集]ではユーザーが作成したModuleのメンテナンス（閲覧／登録／更新／廃止）を行います。

Moduleの記述などに関しては、「 6.1 Moduleの記述」を参照してください。



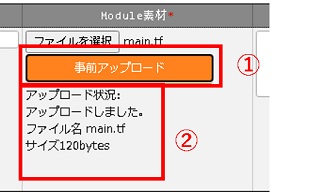
図 5.2.44-1サブメニュー画面（Module素材集）

1. 「登録」-「登録開始」ボタンより、Movement情報の登録を行います。



図 5.2.44-2登録画面（Module素材集）

「登録」の前に、「Module素材」を「事前アップロード（①）」してください。「アップロード状況（②）」にModuleのファイル名が表示されたのを確認してから、「登録」ボタンを押してください。



1. Movement-Module紐付ボタンをクリックすると対象のMovementの「5.2.5 Movement-Module紐付」へ遷移します。



図 5.2.44-3サブメニュー画面（Module素材集）

1. Module素材集の項目一覧は以下のとおりです。

**表 5.2.44-1 項目一覧（Module素材集）**

| **項目** | **説明** | **入力**  **必須** | **入力形式** | **制約事項** |
| --- | --- | --- | --- | --- |
| Module素材名 | ITAで管理するModule素材名を入力します。 | ○ | 手動入力 | 最大長256バイト |
| Module素材 | 作成したModuleファイルをアップロードします。 | ○ | ファイル選択 | 最大サイズ4Gバイト |
| Movement-Module紐付 | 「5.2.5 Movement-Module紐付」へ遷移するボタンです。 | - | - |  |
| 備考 | 自由記述欄です。 | - | 手動入力 | 最大長4000バイト |

1. BackYardの処理でModuleファイル内に定義している変数を抜出します。抜出した変数は「5.2.7代入値自動登録設定」や「5.2.8代入値管理」で具体値の登録が可能になります。

抜出するタイミングはリアルタイムではありませんので「5.2.7代入値自動登録設定」や「5.2.8代入値管理」で変数が扱えるまでに時間がかかる※１場合があります。

**※1 抜出のタイミングは「 7.2メンテナンス方法について」の「③ 起動周期の変更」に記載していますので、そちらをご参照ください。**

### Movement-Module紐付

1. [Movement-Module紐付]では、Movementで実行するModule素材のメンテナンス（閲覧／登録／更新／廃止）を行います。

Movementに対して複数のModule素材を紐付けることが可能です。



**図5.2.55-1サブメニュー画面（Movement-Module紐付）**

1. 「登録」-「登録開始」ボタンより、Movement-Module紐付の登録を行います。



図 5.2.55-2登録画面（Movement-Module紐付）

1. Movementのリンクをクリックすると、「5.2.3 Movement一覧」へ遷移します。

また、Module素材のリンクをクリックすると、「5.2.4 Module素材集」へ遷移します。



**図5.2.55-3サブメニュー画面（Movement-Module紐付）**

1. Movement-Module紐付の項目一覧は以下のとおりです。

**表 5.2.55-1項目一覧（Movement-Module紐付）**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **項目** | **説明** | **入力**  **必須** | **入力形式** | **制約事項** |
| Movement | 「5.2.3 Movement一覧」にて登録したMovementを選択します。 | ○ | リスト選択 | - |
| Module素材 | 「5.2.4 Module素材集」で登録したModule素材を選択します。 | ○ | リスト選択 | - |
| 備考 | 自由記述欄です。 | - | 手動入力 | 最大長4000バイト |

### 変数ネスト管理

1. [変数ネスト管理]では、Module素材集で登録したtfファイルで定義されている変数のタイプがlist,setかつ、その変数の中でlist,set,tuple,objectが定義されている場合、メンバー変数の最大繰返数を閲覧及び更新できます。

本メニューはModule素材集を元にBackYardがレコードを管理するため、登録・廃止・復活はできません。

変数ネストの管理対象については「 8.1Module素材記入例」をご参照ください。

変数ネストの管理フロー例については「 8.2変数ネスト管理フロー例」をご参照ください。



図 5.2.66‑1 サブメニュー画面（変数ネスト管理）

**Module素材と変数ネスト管理登録例**

・Module素材（tfファイル）と登録値

|  |
| --- |
| variable "VAR\_hoge" {  type = list(  object({  IP = string,  NAME = string  })  )  default = [  { “IP” = “127.0.0.1”, “NAME” = “machine\_01” },  { “IP” = “127.0.0.2”, “NAME” = “machine\_02” }  ]  } |

・変数ネスト管理

|  |  |  |  |
| --- | --- | --- | --- |
| **項番** | **変数名** | **メンバー変数** | **最大繰返数** |
| 1 | VAR\_hoge |  | 2 |

1. 「一覧」-「更新」ボタンより最大繰返数の更新を行います。

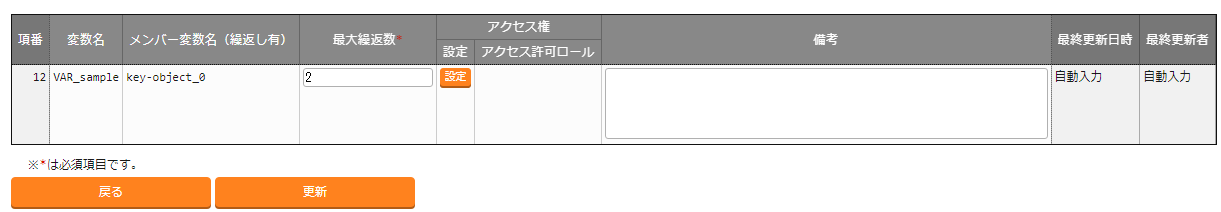


図 5.2.66‑2 更新画面（変数ネスト管理）

1. 変数ネスト管理の項目一覧は以下の通りです。

**表 5.2.56-1項目一覧（変数ネスト管理）**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **項目** | **説明** | **入力**  **必須** | **入力形式** | **制約事項** |
| 変数名 | Movement-Module紐付で登録した資材で使用している変数が表示されます。 | - | 入力不可 |  |
| メンバー変数名（繰返し有） | 変数ネスト管理対象がメンバー変数である場合、メンバー変数名が表示されます。メンバー変数名は各階層の変数を「.」で連結して表示します。 | - | 入力不可 |  |
| 最大繰返数 | 配列の最大繰返数を1～99,999,999の範囲で入力します。  初期値はtfファイルのdefaultに記載されている値から取得した繰返数が設定されます。  tfファイルにdefaultの記載がない場合、1が設定されます。  最終更新者が「Terraform変数更新プロシージャ」でない場合はModule素材の更新により値が変更されることはありません。 | ○ | 手動入力 | 入力値 1～99,999,999 |
| 備考 | 自由記述欄です。 | - | 手動入力 | 最大長4000バイト |

※初期登録および繰返数の更新はリアルタイムではないので、「5.2.7代入値自動登録設定」や「5.2.8代入値管理」で変数が扱えるまでに時間がかかる場合があります。

初期登録および繰返数の更新タイミングは「 7.2メンテナンス方法について」に記載していますので、そちらをご参照ください。

1. アクセス許可ロール

変数ネスト管理に設定されるアクセス許可ロールは、該当の変数が定義されているModule素材集のアクセス許可ロールが設定されます。

### 代入値自動登録設定

1. [代入値自動登録設定]では、メニュー作成機能で作成したパラメータシート（オペレーションあり）と、Movementの変数を紐付けます。登録した情報はBackYardの処理により代入値管理に反映されます。

6.2 BackYard コンテンツ(2)代入値自動登録設定に反映ルールを記載しています。



**図5.2.67-1サブメニュー画面（代入値自動登録）**

1. 「登録」-「登録開始」ボタンより、代入値管理を行います。



**図 5.2.67-2登録画面（代入値自動登録）**

1. メニューIDまたは、メニュー名のリンクをクリックすると、対象のメニューへ遷移します。



**図5.2.67-3サブメニュー画面（代入値自動登録）**

1. 登録画面の項目一覧は以下のとおりです。

**表5.2.7-1　項目一覧（代入値自動登録）**

| **カラム** | | **説明** | **入力必須** | **入力形式** | **制約事項** |
| --- | --- | --- | --- | --- | --- |
| メニューグループ：メニュー | | メニュー作成機能で作成したパラメータシート（オペレーションあり）が表示 されます。 該当のパラメータシートを選択します。 | ○ | リスト選択 |  |
| 項目 | | 選択したパラメータシートの項目が表示されます。 対象の項目を選択します。 | ○ | リスト選択 |  |
| 登録方式 | | Value 型：項目の設定値を紐付けた変数の具体値とする場合に選択します。  Key 型：項目の名称を紐付けた変数の具体値とする場合に選択します。  項目の設定値が空白の場合は紐付け対象外となり  ます。  Key-Value 型：項目の名称(Key)と設定値(Value)を  紐付けた変数の具体値とする場合に選択します。 | ○ | リスト選択 |  |
| Movement | | Movement 一覧で登録した Movement が表示されます。Movement を選択します。 | ○ | リスト選択 |  |
| Key  情報 | 変数名 | Movement-Module紐付で登録した資材で使用している変数が表示されます。  Key型で具体値に紐付けたい変数を選択します。 | ○ | リスト選択 | 登録方式でKey型  またはKey-Value型を  選択した場合は必須 |
| メンバー変数 | メンバー変数を持つ変数名を選択した場合に  メンバー変数が表示されます。  メンバー変数を選択します。 | 変数名により変動 | リスト選択 |  |
| 代入順序 | 複数具体値が設定できる変数名の場合のみ必須入力になります。  具体値の代入順序（1～）を入力します。入力値に従い昇順で代入されます。 | 変数名により変動 | 手動入力 | ブランク  または、  正の整数 |
| Value  情報 | 変数名 | Movement-Module紐付で登録した資材で使用している変数が表示されます。  Value型で具体値に紐付けたい変数を選択します。 | ○ | リスト選択 | 登録方式でValue型  またはKey-Value型を  選択した場合は必須 |
| HCL設定 | 「OFF」または「ON」を選択します。  BackYardの処理により代入値管理に反映する際、選択した値が引き継がれます。 | ○ | リスト選択 | 選択した変数名がmap型の場合はONで設定する必要があります。  オペレーション、Movement、変数名が一致している場合、HCL設定はONまたはOFFに統一してください。 |
| メンバー変数 | メンバー変数を持つ変数名を選択した場合に  メンバー変数が表示されます。  メンバー変数を選択します。 | 変数名により変動 | リスト選択 |  |
| 代入順序 | 複数具体値が設定できる変数名の場合のみ必須入力になります。  具体値の代入順序（1～）を入力します。入力値に従い昇順で代入されます。 | 変数名により変動 | 手動入力 | ブランク  または、  正の整数 |
| NULL連携 | | パラメータシートの具体値がNULL(空白)の場合に、代入値管理への登録をNULL(空白)の値で行うか設定します。  ・「有効」の場合、パラメータシートの値がどのような値でも代入値管理への登録が行われます。  ・「無効」の場合、パラメータシートに値が入っている場合のみ代入値管理への登録が行われます。  ・空白の場合、インターフェース情報の「NULL連携」の値が適用されます。 | - | リスト選択 | - |
| 備考 | | 自由記述欄です。 | - | 手動入力 | 最大長4000バイト |

※登録方式でKey型を選択した場合、代入値管理に反映する際にHCL設定はOFFで設定されます。

※メンバー変数を設定する場合は、同じ変数内のメンバー変数の具体値も全て設定してください。

代入値を設定しなかった他のメンバー変数でもデフォルト値が使用されることはありません。

例

・tfファイルと登録値

|  |
| --- |
| variable "VAR\_hoge" {  type = object({  IP = string  NAME = string  })  default = {  IP = “127.0.0.1”  NAME = “machine01”  }  } |

・代入値例（代入値自動登録設定/代入値管理）

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **項番** | **変数名** | **メンバー変数** | **代入順序** | **具体値** |
| 1 | VAR\_hoge | IP | 入力不要 | 192.168.0.1 |
| 2 | VAR\_hoge | NAME | 入力不要 |  |

ここの値を設定する

必要がある

### 代入値管理

1. [代入値管理]では、オペレーションごとに、対象のMovementで利用されるModule内の変数に代入する具体値をメンテナンス（閲覧／登録／更新／廃止）できます。



**図5.2.88-1サブメニュー画面（代入値管理）**

1. 「登録」-「登録開始」ボタンより、代入値管理を行います。



図 5.2.88-2登録画面（代入値管理）

代入値管理の変数は、「5.2.4 Module素材集」で登録されたファイルの情報から反映されます。

**※ 反映のタイミングは「 7.2メンテナンス方法について」の「③起動周期の変更」に記載していますので、そちらをご参照ください**

1. 代入値管理に登録した変数は、作業実行時に生成されるterraform.tfvarsファイルに「変数名」が「Key」、「具体値」が「Value」として記載され、作業実行で使用されます。

「Secure設定」を「ON」にしていた場合は作業実行時に生成されるsecure.tfvarsファイルに「変数名」が「Key」、「具体値」が「Value」として記載され、作業実行で使用されます。secure.tfvarsファイルは「5.2.10作業状態確認」「5.2.11作業管理」で取得できる投入データの中に格納されません。

1. 代入値管理の項目一覧は以下のとおりです。

**表5.2.8-1　項目一覧（代入値管理）**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **項目** | **説明** | **入力**  **必須** | **入力形式** | **制約事項** |
| オペレーション | 対象のオペレーションを選択します。 | ○ | リスト選択 | - |
| Movement | 対象のMovementを選択します。 | ○ | リスト選択 | - |
| 変数名 | Movement-Module紐付にて登録されているModule素材の中から、選択されたMovementにアタッチしている変数名が表示されます。変数を選択します。 | ○ | リスト選択 | - |
| HCL設定 | 「OFF」または「ON」を選択します。  変数に文字列以外の値を設定する際などに「ON」を設定します。 | ○ | リスト選択 | 選択した変数名がmap型の場合はONで設定する必要があります。  オペレーション、Movement、変数名が一致している場合、HCL設定はONまたはOFFに統一してください。 |
| メンバー変数 | メンバー変数を持つ変数名を選択した場合に  メンバー変数が表示されます。  メンバー変数を選択します。 | 変数名により変動 | リスト選択 |  |
| 代入順序 | 複数具体値が設定できる変数名の場合のみ必須入力になります。  具体値の代入順序（1～）を入力します。入力値に従い昇順で代入されます。 | 変数名により変動 | 手動入力 | ブランク  または、  正の整数 |
| デフォルト値 | default内で変数に紐づいている具体値が表示されます。 | - |  |  |
| Secure設定 | 「OFF」または「ON」を選択します。  「ON」を選択した場合、具体値を暗号化しITA上で表示させないようにします。  また「ON」にした対象は「5.2.10作業状態確認」「5.2.11作業管理」で取得できる投入データの中に格納されません。 | ○ | リスト選択 | Terraformが出力するPlanやApplyのログ上で値を非表示にする場合は、登録したModule素材のvariableブロックの中で『sensitive = true』を指定する必要があります。 |
| 具体値 | オペレーション/Movementで使用する変数の具体値を入力します。 | ○ | 手動入力 | 最大長8192バイト |
| 備考 | 自由記述欄です。 | - | 手動入力 | 最大長4000バイト |

### 作業実行

1. 作業の実行を指示します。Movement一覧、オペレーション一覧からそれぞれラジオボタンで選択し、実行ボタンを押すと、「5.2.10作業状態確認」に遷移し、実行されます。



**図 5.2.99-1 サブメニュー画面（作業実行）**

1. **予約日時の指定**

「予約日時」を入力することで、実行およびPlan確認を予約することができます。

「予約日時」には、未来の日時のみ登録可能です。

1. **Movementの指定**

「5.2.3 Movement一覧」で登録したMovementを選択します。

1. **オペレーションの指定**

「5.1.1オペレーション一覧」で登録したオペレーションを選択します。

1. **実行**

「実行」ボタンをクリックすると、「5.2.10作業状態確認」に遷移し、作業が実行されます。

Plan完了後にApplyが自動で実行されます。

1. **Plan確認**

「Plan確認」ボタンをクリックすると、「実行」ボタンをクリックした場合同様に作業実行が開始されますが、Planのみを実行し、Applyは実行されません。

1. Outputブロックを含むModuleがConductorから実行された場合、Outputブロックに書かれた内容がデータリレイストレージパス（Conductor実行時、各Movementで共有するディレクトリを、ITAサーバから見たディレクトリパス）にjson形式ファイルで保存されます。

このファイルを使用することにより、同一Conductorの別のMovementでTerraformが出力した値を使用することができます。

ファイルパス

[データリレイストレージパス]/[ConductorインスタンスID]/terraform\_output\_[作業No].json

例：/exastro/data\_relay\_storage/conductor/0000000001/terraform\_output\_0000000001.json

データリレイストレージパス・・・[Conductor]-[Conductorインターフェース情報]メニューの[データリレイストレージパス]

ConductorインスタンスID・・・[Conductor]-[Conductor作業一覧]メニューの[ConductorインスタンスID]（左0埋め10桁）

作業No.・・・[Terraform]-[作業管理]メニューの[作業No.]（左0埋め10桁）

記述例

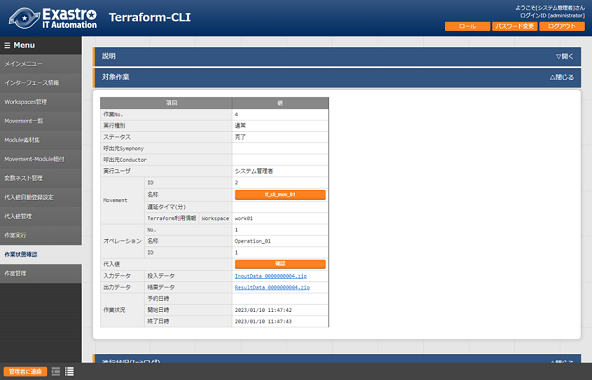
|  |
| --- |
| variable "VAR\_sample" {  type = string  default = "sample\_string"  }  output "output\_sample" {  value = "${var.VAR\_sample}"  } |

出力例

|  |
| --- |
| {  "output\_sample": "sample\_string"  } |

### 作業状態確認

1. 作業の実行状態を監視します。



**図 5.2.100-1サブメニュー画面（作業状態確認）**

1. **実行状態表示**実行状況に即し、ステータスが表示されます。

「実行種別」にはPlan確認の場合には「Plan確認」、Workspaceごとに構成・管理されたリソースの削除（「5.2.2 Workspaces管理」から実行されます。）の場合は「リソース削除」、それ以外の場合には「通常」が入ります。

「進行状況(Initログ)」「進行状況(Planログ)」「進行状況(Applyログ)」にはTerraformで実行するコマンドterraform init / terraform plan / terraform applyの各実行状況のログが表示されます。

ステータスが想定外エラーで終了した場合、「5.2.1インターフェース情報」の登録不備や、その他のWebコンテンツの登録不備が原因であれば、「進行状況(エラーログ)」にメッセージが表示されます。

それ以外のエラーの場合は「進行状況(エラーログ)」にメッセージが表示されません。この場合は、プロセスのログにエラー情報が記録されます。必要に応じてプロセスのログを確認ください。

※プロセスのログは「 7.2メンテナンス方法について」の「④ログファイル名」に記載していますので、そちらをご参照ください。

「呼出元Symphony」には、どのSymphonyから実行されたかを表示します。Terraform CLIドライバから直接実行した場合やConductorから実行した場合は空欄になります。

「呼出元 Conductor」には、どのConductorから実行されたかを表示します。Terraform CLIドライバから直接実行した場合やSymphonyから実行した場合は空欄になります。

「実行ユーザ」には、作業実行メニューより「実行」ボタンを押下した際のログインユーザが表示されます。

※「実行種別」が「リソース削除」の場合は下記の項目が設定されません。

・呼出元Symphony

・呼出元Conductor

・Movement（ID、名称、遅延データ（分））

・オペレーション（No.、名称、ID）

・代入値

・入力データ（投入データ）

1. **代入値確認**

「確認」ボタンで「5.2.8代入値管理」が表示され、作業対象のオペレーションとMovementに絞り込んだ代入値が表示されます。

1. **緊急停止/予約取り消し**

「緊急停止」ボタンで構築作業を停止させることができます。

また、実行前の「予約実行」の作業の場合は、「予約取消」ボタンが表示されます。「予約取消」ボタンで予約実行が取り消せます。

1. **ログ検索**実行ログ、エラーログは、フィルタリングができます。各ログのフィルタのテキストボックスに検索したい文字列を入力し、「該当行のみ表示」のチェックボックスをチェックすることで該当する行だけが表示されます。  
   実行ログ、エラーログのリフレッシュ表示間隔と最大表示行数を、「5.2.1インターフェース情報」の「状態監視周期（単位ミリ秒）」と「進行状態表示行数」で設定できます。
2. **投入データ**実行したModule素材および設定した代入値の一覧をjson形式で取得したファイルを格納したzip形式ファイルをダウンロードすることができます。

格納されているファイルは以下の通りです。

**表 5.2.100-1 投入データ格納ファイル**

|  |  |
| --- | --- |
| **ファイル名** | **説明** |
| (投入したModule素材ファイル名) | 投入したModule素材ファイルがzipファイルの直下にすべて格納されます。 |
| terraform.tfvars | 設定した各代入値についての「変数名(key)」「具体値(value)」を記載したファイルです。  Secure設定がONの対象は記載されません。 |

1. **結果データ**実行ログ、エラーログおよび、terraformコマンドが生成したファイルを格納したzip形式ファイルをダウンロードすることができます。

格納されているファイルは以下の通りです。

**表 5.2.100-2 結果データ格納ファイル**

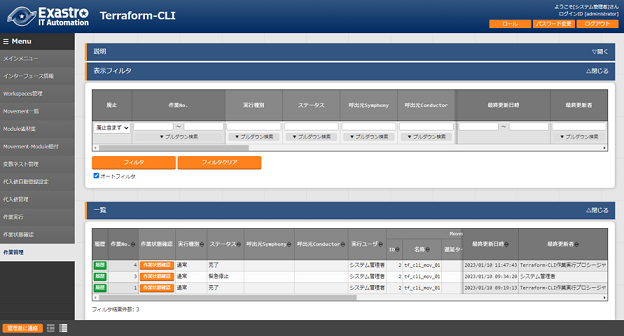
|  |  |
| --- | --- |
| **ファイル名** | **説明** |
| Init.log | 進行状況(Initログ)に出力された内容を記載したlogファイルです。 |
| plan.log | 進行状況(Planログ)に出力された内容を記載したlogファイルです。 |
| apply.log | 進行状況(Applyログ)に出力された内容を記載したlogファイルです。 |
| error.log | 進行状況(エラーログ)に出力された内容を記載したlogファイルです。 |
| result.txt | 作業実行時にバックヤードが利用する進行状況を記録するファイルです。 |
| .terraform.lock.hcl | Terraformが生成したファイルです。providerやmoduleの情報が記載されます。 |
| terraform.tfstate | Terraformが生成したstateファイルです。  暗号化された状態で保存されています。 |
| terraform.tfstate.backup | Terraformが生成したstateファイルのバックアップです。  暗号化された状態で保存されています。 |

### 作業管理

1. 作業の履歴を閲覧できます。

条件を指定し「フィルタ」ボタンをクリックすると、作業一覧テーブルを表示します。

「作業状態確認」ボタンで、「5.2.10作業状態確認」に遷移し、実行状態の詳細を見ることができます。



**図5.2.111-1サブメニュー画面（作業管理）**

# 構築コード記述方法

Terraform-CLI-driverでModuleの記述について説明します。

## Moduleの記述

Moduleファイルは、HCL（HashiCorp Configuration Language）というHashiCorp社独自の言語により記述します。

HCLの詳細については、Terraformの製品マニュアルを参照してください。

## BackYard コンテンツ

1. **変数自動登録**

「5.2.4 Module素材集」に登録したModule素材ファイルから変数を抽出します。

変数の抽出ルールについては「 2.1変数の種類」をご参照ください。

なお、取出すタイミングは「Terraform変数自動登録」プロセスの起動周期に依存します。

1. **代入値自動登録設定**

連携対象としたパラメータシートのオペレーションの項目の設定値と紐付けたMovementと変数の情報を代入値管理に反映します。

なお、取出すタイミングは「Terraform自動登録設定」プロセスの起動周期に依存します。

代入値管理は複数の操作者が更新を行います。最終更新者が他操作者の場合は反映処理をしません。

代入値自動登録設定のデータを反映したい場合は、代入値管理で該当レコードを廃止にする操作を行ってください。

代入値管理への反映ルールを以下に記載します。

1. **代入値自動登録設定に登録されている情報を代入値管理へ反映時**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **代入値管理**  **の状態** | **該当レコード**  **なし** | **該当レコードあり** | | | **該当レコード**  **廃止中** |
| **=具体値** | **≠具体値** | |
| **最終更新者** | |
| **BackYard処理** | **他操作者** |
| **代入値管理**  **への反映** | 新規レコード  追加 | - | 該当レコードの  具体値更新 | - | 廃止レコード  復活 |

※ 該当レコード：オペレーション＋Movement+変数名+HCL設定+アクセス権が同一のレコードの意

1. **代入値自動登録設定に登録されていない情報（代入値管理のみに登録）を代入値管理へ反映**

|  |  |  |
| --- | --- | --- |
| **代入値管理 の状態** | **該当レコードあり** | |
| **最終更新者** | |
| **BackYard処理** | **他操作者** |
| **代入値管理への反映** | 該当レコード廃止 | - |

1. **HCL設定について**

代入値自動登録に設定した「HCL設定」の値は代入値管理に反映する際に同じ値が設定されます。

1. **Secure設定について**

連携対象としたパラメータシートの項目が「パスワード」の場合は代入値管理に反映する際に「Secure設定」がONで設定されます。それ以外の場合はOFFで設定されます。

1. **アクセス許可ロールについて**

連連携対象としたパラメータシートのレコードに設定されたオペレーションのアクセス許可ロールと、代入値自動登録のレコードに設定されたMovementのアクセス許可ロールを参照し、一致しているすべてのアクセス許可ロールが代入値管理に反映する際に設定されます。

どちらもアクセス許可ロールが設定されていない場合（空欄の場合）は、代入値管理に反映する際も空欄が設定されます。

また、一致するアクセス許可ロールが一つも無い場合は代入値管理にレコードが作成されません。

# 運用操作

本機能を活用する操作は、クライアントPCのブラウザ画面からのユーザー利用による入力だけでなく、システム運用・保守による操作もあります。用意している運用・保守の操作は次のとおりです。

## メンテナンス

Terraform-CLI-driverのプロセスの開始/停止/再起動に必要なファイルは以下となります。

|  |  |
| --- | --- |
| **説明** | **対象ファイル名** |
| Terraform作業実行監視  未実行作業の実行を行う。 | ky\_terraform\_cli\_execute-workflow.service |
| Terraform変数自動登録  アップロードしたModule素材から変数・メンバー変数・最大繰返数の取出しを行う。  最大繰返数をメンバー変数に反映する。 | ky\_terraform\_cli\_varsautolistup-workflow.service |
| Terraform自動登録設定  代入値自動登録に設定された情報を代入値管理に反映を行う。 | ky\_terraform\_cli\_valautosetup-workflow.service |

対象ファイルは「/usr/lib/systemd/system」に格納されています。

プロセス起動/停止/再起動の方法は次の通りです。

root権限でコマンドを実行してください。

1. プロセス起動

# systemctl start ky\_terraform\_cli\_execute-workflow.service

1. プロセス停止

# systemctl stop ky\_terraform\_cli\_execute-workflow.service

1. プロセス再起動

# systemctl restart ky\_terraform\_cli\_execute-workflow.service

各対象ファイル名に置き換えて起動/停止/再起動を行ってください。

## メンテナンス方法について

①　NORMALレベルへの変更

　以下のファイルの8行目「DEBUG」を「NORMAL」に書き換えます。

　ログレベル設定ファイル： <インストールディレクトリ>/ita-root/confs/backyardconfs/ita\_env

②　DEBUGレベルへの変更

　以下のファイルの8行目「NORMAL」を「DEBUG」に書き換えます。

　ログレベル設定ファイル： <インストールディレクトリ>/ita-root/confs/backyardconfs/ita\_env

③ 起動周期の変更

　各対象ファイルのExecStartの5番目のパラメータを変更します。（単位:秒）

　例外を除き起動周期はデフォルト値の使用をしてください。

ExecStart=/bin/sh ${ITA\_ROOT\_DIR}/backyards/common/ky\_loopcall-php-procedure.sh /bin/php /bin/php ${ITA\_ROOT\_DIR}/backyards/terraform\_driver/ky\_terraform\_execute-workflow.php ${ITA\_ROOT\_DIR}/logs/backyardlogs **5** ${ITA\_LOG\_LEVEL} > /dev/null 2>&1

書き換え後、**プロセス再起動(restart)後に有効になります。**

④ ログファイル名

|  |  |
| --- | --- |
| **プロセス名** | **ログファイル名** |
| ky\_terraform\_cli\_execute-workflow | ky\_terraform\_cli\_execute-workflow\_YYYYMMDD.log |
| ky\_terraform\_cli\_execute-workflow | ky\_terraform\_cli\_execute-child-workflow\_YYYYMMDD.log |
| ky\_terraform\_cli\_varsautolistup-workflow | ky\_terraform\_cli\_varsautolistup-workflow\_YYYYMMDD.log |
| ky\_terraform\_cli\_valautosetup-workflow | ky\_terraform\_cli\_valautosetup-workflow\_YYYYMMDD.log |

ログファイルの出力先： <インストールディレクトリ>/ita-root/logs/backyardlogs

# 付録

## Module素材記入例・登録例

Module素材の記入例と登録例を「 4.1 Terraform作業フロー」のフロー番号と照らし合わせて記載します。

* + - 1. シンプルなパターン
  1. string型

|  |
| --- |
| variable "VAR\_hoge" {  type = string  default = "def-string"  } |

**代入値自動登録設定/代入値管理**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **項番** | **変数名** | **メンバー変数** | **代入順序** | **具体値** |
| 1 | VAR\_hoge | 入力不要 | 入力不要 | sample\_str |

**TerraformのVariablesに登録される値**

|  |  |
| --- | --- |
| **Key** | **Value** |
| VAR\_hoge | sample\_str |

**⑧ 変数値の設定**

**⑤ Module素材の登録**

**⑨ 作業実行**

* 1. number型

|  |
| --- |
| variable "VAR\_hoge" {  type = number  default = 2022  } |

**代入値自動登録設定/代入値管理**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **項番** | **変数名** | **メンバー変数** | **代入順序** | **具体値** |
| 1 | VAR\_hoge | 入力不要 | 入力不要 | 2023 |

**TerraformのVariablesに登録される値**

|  |  |
| --- | --- |
| **Key** | **Value** |
| VAR\_hoge | 2023 |

**⑧ 変数値の設定**

**⑤ Module素材の登録**

**⑨ 作業実行**

* 1. bool型

|  |
| --- |
| variable "VAR\_hoge" {  type = bool  default = true  } |

**代入値自動登録設定/代入値管理**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **項番** | **変数名** | **メンバー変数** | **代入順序** | **具体値** |
| 1 | VAR\_hoge | 入力不要 | 入力不要 | false |

**TerraformのVariablesに登録される値**

|  |  |
| --- | --- |
| **Key** | **Value** |
| VAR\_hoge | false |

**⑧ 変数値の設定**

**⑤ Module素材の登録**

**⑨ 作業実行**

* 1. list型

|  |
| --- |
| variable "VAR\_hoge" {  type = list(string)  default = [“aaa”, “bbb”, “ccc”]  } |

**代入値自動登録設定/代入値管理**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **項番** | **変数名** | **HCL設定** | **メンバー変数** | **代入順序** | **具体値** |
| 1 | VAR\_hoge | OFF | 入力不要 | 1 | あああ |
| 2 | VAR\_hoge | OFF | 入力不要 | 2 | いいい |

**TerraformのVariablesに登録される値**

|  |  |
| --- | --- |
| **Key** | **Value** |
| VAR\_hoge | [“あああ”, “いいい”] |

**⑧ 変数値の設定**

**⑤ Module素材の登録**

**⑨ 作業実行**

* 1. set型

|  |
| --- |
| variable "VAR\_hoge" {  type = set(string)  default = [“aaa”, “bbb”, “ccc”]  } |

**代入値自動登録設定/代入値管理**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **項番** | **変数名** | **HCL設定** | **メンバー変数** | **代入順序** | **具体値** |
| 1 | VAR\_hoge | OFF | 入力不要 | 1 | あああ |
| 2 | VAR\_hoge | OFF | 入力不要 | 2 | いいい |

**TerraformのVariablesに登録される値**

|  |  |
| --- | --- |
| **Key** | **Value** |
| VAR\_hoge | [“あああ”, “いいい”] |

**⑧ 変数値の設定**

**⑤ Module素材の登録**

**⑨ 作業実行**

* 1. tuple型

|  |
| --- |
| variable "VAR\_hoge" {  type = tuple([string, number])  default = [“aaa”, 2022]  } |

**代入値自動登録設定/代入値管理**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **項番** | **変数名** | **HCL設定** | **メンバー変数** | **代入順序** | **具体値** |
| 1 | VAR\_hoge | OFF | [0] | 入力不要 | あああ |
| 2 | VAR\_hoge | OFF | [1] | 入力不要 | 2023 |

**TerraformのVariablesに登録される値**

|  |  |
| --- | --- |
| **Key** | **Value** |
| VAR\_hoge | [“あああ”, 2023] |

**⑧ 変数値の設定**

**⑤ Module素材の登録**

**⑨ 作業実行**

* 1. map型

map型はHCL設定をONにする必要があります。

|  |
| --- |
| variable "VAR\_hoge" {  type = map(string)  default = {  “key” = “value”  }  } |

**代入値自動登録設定/代入値管理**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **項番** | **変数名** | **HCL設定** | **メンバー変数** | **代入順序** | **具体値** |
| 1 | VAR\_hoge | ON | 入力不要 | 入力不要 | { “aaa” = “bbb”} |

**TerraformのVariablesに登録される値**

|  |  |
| --- | --- |
| **Key** | **Value** |
| VAR\_hoge | { “aaa” = “bbb”} |

**⑧ 変数値の設定**

**⑤ Module素材の登録**

**⑨ 作業実行**

* 1. object型

|  |
| --- |
| variable "VAR\_hoge" {  type = object({  IP = string,  NAME = string  })  default = {  “IP” = “127.0.0.1”,  “NAME” = “machine01”  }  } |

**代入値自動登録設定/代入値管理**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **項番** | **変数名** | **HCL設定** | **メンバー変数** | **代入順序** | **具体値** |
| 1 | VAR\_hoge | OFF | IP | 入力不要 | 192.168.0.1 |
| 2 | VAR\_hoge | OFF | NAME | 入力不要 | my\_machine |

**TerraformのVariablesに登録される値**

|  |  |
| --- | --- |
| **Key** | **Value** |
| VAR\_hoge | {  “IP” = “192.168.0.1”,  “NAME” = “my\_machine”  } |

**⑧ 変数値の設定**

**⑤ Module素材の登録**

**⑨ 作業実行**

* 1. any型

|  |
| --- |
| variable "VAR\_hoge" {  type = any  default = “def-any”  } |

**代入値自動登録設定/代入値管理**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **項番** | **変数名** | **HCL設定** | **メンバー変数** | **代入順序** | **具体値** |
| 1 | VAR\_hoge | OFF | key | 入力不要 | aaa |

**TerraformのVariablesに登録される値**

|  |  |
| --- | --- |
| **Key** | **Value** |
| VAR\_hoge | aaa |

**⑧ 変数値の設定**

**⑤ Module素材の登録**

**⑨ 作業実行**

* 1. typeの記載がない

|  |
| --- |
| variable "VAR\_hoge" {  default = “def-string”  } |

**代入値自動登録設定/代入値管理**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **項番** | **変数名** | **HCL設定** | **メンバー変数** | **代入順序** | **具体値** |
| 1 | VAR\_hoge | OFF | 入力不要 | 入力不要 | aaa |

**TerraformのVariablesに登録される値**

|  |  |
| --- | --- |
| **Key** | **Value** |
| VAR\_hoge | aaa |

**⑧ 変数値の設定**

**⑤ Module素材の登録**

**⑨ 作業実行**

* + - 1. 複雑なパターン

1. list型配下のlist

|  |
| --- |
| variable "VAR\_hoge" {  type = list(list(string))  default = [  [“a”, “b”, “c”],  [“d”, “e”, “f”]  ]  } |

|  |  |  |  |
| --- | --- | --- | --- |
| **項番** | **変数名** | **メンバー変数名（繰返し有）** | **最大繰返数** |
| 1 | VAR\_hoge |  | 2 |

**代入値自動登録設定/代入値管理**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **項番** | **変数名** | **HCL設定** | **メンバー変数** | **代入順序** | **具体値** |
| 1 | VAR\_hoge | OFF | [0] | 1 | あああ |
| 2 | VAR\_hoge | OFF | [0] | 2 | いいい |
| 3 | VAR\_hoge | OFF | [1] | 1 | かかか |

**TerraformのVariablesに登録される値**

|  |  |
| --- | --- |
| **Key** | **Value** |
| VAR\_hoge | [  [“あああ”, “いいい”],  [“かかか”]  ] |

**⑧ 変数値の設定**

**⑤ Module素材の登録**

**⑨ 作業実行**

**⑦ 最大繰返数の設定**

1. list型配下のobject

|  |
| --- |
| variable "VAR\_hoge" {  type = list(  object({  NAME = string  AGE = number  })  )  default = [  { “NAME” = “田中”, “AGE” = 30 },  { “NAME” = “山本”, ” AGE” = 26 }  ]  } |

**変数ネスト管理**

|  |  |  |  |
| --- | --- | --- | --- |
| **項番** | **変数名** | **メンバー変数名（繰返し有）** | **最大繰返数** |
| 1 | VAR\_hoge |  | 2 |

**代入値自動登録設定/代入値管理**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **項番** | **変数名** | **HCL設定** | **メンバー変数** | **代入順序** | **具体値** |
| 1 | VAR\_hoge | OFF | [0].NAME | 入力不要 | 本田 |
| 2 | VAR\_hoge | OFF | [0].AGE | 入力不要 | 20 |
| 3 | VAR\_hoge | OFF | [1].NAME | 入力不要 | 荻窪 |
| 4 | VAR\_hoge | OFF | [1].AGE | 入力不要 | 50 |

**TerraformのVariablesに登録される値**

|  |  |
| --- | --- |
| **Key** | **Value** |
| VAR\_hoge | [  {“NAME” = “本田”, “AGE” = 20}  {“NAME” = “荻窪”, “AGE” = 50}  ] |

**⑧ 変数値の設定**

**⑤ Module素材の登録**

**⑨ 作業実行**

**⑦ 最大繰返数の設定**

1. object配下のlist配下のobject

|  |
| --- |
| variable "VAR\_hoge" {  type = object({  FRUIT = list(object{  NAME = string, PRICE = number  }),  VEGETABLE = l ist(object{  NAME = string, PRICE = number  })  })  default = {  FRUIT = [  { NAME = “りんご”, PRICE = 120 },  { NAME = “みかん”, PRICE = 80 }  ],  VEGETABLE = [  { NAME = “なす”, PRICE = 100 },  { NAME = “トマト”, PRICE = 200 }  ]  }  } |

**変数ネスト管理**

|  |  |  |  |
| --- | --- | --- | --- |
| **項番** | **変数名** | **メンバー変数名（繰返し有）** | **最大繰返数** |
| 1 | VAR\_hoge | FRUIT | 2 |
| 2 | VAR\_hoge | VEGETABLE | 2 |

**代入値自動登録設定/代入値管理**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **項番** | **変数名** | **HCL設定** | **メンバー変数** | **代入順序** | **具体値** |
| 1 | VAR\_hoge | OFF | FRUIT.[0].NAME | 入力不要 | キウイ |
| 2 | VAR\_hoge | OFF | FRUIT.[0].PRICE | 入力不要 | 200 |
| 3 | VAR\_hoge | OFF | FRUIT.[1].NAME | 入力不要 | ぶどう |
| 4 | VAR\_hoge | OFF | FRUIT.[1].PRICE | 入力不要 | 1000 |
| 5 | VAR\_hoge | OFF | VEGETABLE.[0].NAME | 入力不要 | 白菜 |
| 6 | VAR\_hoge | OFF | VEGETABLE.[0].PRICE | 入力不要 | 100 |
| 7 | VAR\_hoge | OFF | VEGETABLE.[1].NAME | 入力不要 | キャベツ |
| 8 | VAR\_hoge | OFF | VEGETABLE.[1].PRICE | 入力不要 | 110 |

**⑧ 変数値の設定**

**⑤ Module素材の登録**

**⑦ 最大繰返数の設定**

**TerraformのVariablesに登録される値**

|  |  |
| --- | --- |
| **Key** | **Value** |
| VAR\_hoge | {  FRUIT = [  { NAME = “キウイ”, PRICE = 200 },  { NAME = “ぶどう”, PRICE = 1000 }  ],  VEGETABLE = [  { NAME = “白菜”, PRICE = 100 },  { NAME = “キャベツ”, PRICE = 110 }  ]  } |

**⑨ 作業実行**

* + - 1. 特殊な型
  1. list型配下のmap型

map型を含む場合はHCL設定をONにする必要があります。

|  |
| --- |
| variable "VAR\_hoge" {  type = list(map(string))  default = [{  “key” = “value”  }]  } |

**代入値自動登録設定/代入値管理**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **項番** | **変数名** | **HCL設定** | **メンバー変数** | **代入順序** | **具体値** |
| 1 | VAR\_hoge | ON | 入力不要 | 入力不要 | [{ “aaa” = “bbb”}] |

**TerraformのVariablesに登録される値**

|  |  |
| --- | --- |
| **Key** | **Value** |
| VAR\_hoge | [{ “aaa” = “bbb”}] |

**⑧ 変数値の設定**

**⑤ Module素材の登録**

**⑨ 作業実行**

## 変数ネスト管理フロー例

変数ネスト管理の操作例を「 4.1 Terraform作業フロー」のフロー番号と照らし合わせて記載します。

* + - 1. 最大繰返数を増加させる

|  |
| --- |
| variable "VAR\_hoge" {  type = list(object({ IP = string, NAME = string }))  default = [  { “IP” = “127.0.0.1”, NAME = “machine01”},  { “IP” = “127.0.0.2”, NAME = “machine02”}  ],  } |

**変数ネスト管理（登録時の値）**

|  |  |  |  |
| --- | --- | --- | --- |
| **項番** | **変数名** | **メンバー変数名（繰返し有）** | **最大繰返数** |
| 1 | VAR\_hoge |  | 2 |

**変数ネスト管理（更新）**

|  |  |  |  |
| --- | --- | --- | --- |
| **項番** | **変数名** | **メンバー変数名（繰返し有）** | **最大繰返数** |
| 1 | VAR\_hoge |  | 3 |

**代入値自動登録設定/代入値管理**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **項番** | **変数名** | **HCL設定** | **メンバー変数** | **代入順序** | **具体値** |
| 1 | VAR\_hoge | OFF | [0].IP | 入力不要 | 192.168.1.1 |
| 2 | VAR\_hoge | OFF | [0].NAME | 入力不要 | yamamoto |
| 3 | VAR\_hoge | OFF | [1].IP | 入力不要 | 192.168.1.2 |
| 4 | VAR\_hoge | OFF | [1].NAME | 入力不要 | suzuki |
| 5 | VAR\_hoge | OFF | [2].IP | 入力不要 | 192.168.1.3 |
| 6 | VAR\_hoge | OFF | [2].NAME | 入力不要 | tanaka |

**⑧ 変数値の設定**

**⑤ Module素材の登録**

変数ネスト管理更新により

追加されたメンバー変数

**⑦ 最大繰返数の設定**

**TerraformのVariablesに登録される値**

|  |  |
| --- | --- |
| **Key** | **Value** |
| VAR\_hoge | [  { “IP” = “192.168.1.1”, “NAME” = “yamamoto” },  { “IP” = “192.168.1.2”, “NAME” = “suzuki” },  { “IP” = “192.168.1.3”, “NAME” = “tanaka” }  ] |

**⑨ 作業実行**

**⑦ 最大繰返数の設定**

* + - 1. 最大繰返数を減少させる

|  |
| --- |
| variable "VAR\_hoge" {  type = list(object({ IP = string, NAME = string }))  default = [  { “IP” = “127.0.0.1”, NAME = “machine01”},  { “IP” = “127.0.0.2”, NAME = “machine02”},  { “IP” = “127.0.0.3”, NAME = “machine03”}  ],  } |

**変数ネスト管理（登録時の値）**

|  |  |  |  |
| --- | --- | --- | --- |
| **項番** | **変数名** | **メンバー変数名（繰返し有）** | **最大繰返数** |
| 1 | VAR\_hoge |  | 3 |

**変数ネスト管理（更新）**

|  |  |  |  |
| --- | --- | --- | --- |
| **項番** | **変数名** | **メンバー変数名（繰返し有）** | **最大繰返数** |
| 1 | VAR\_hoge |  | 2 |

**⑤ Module素材の登録**

**代入値自動登録設定/代入値管理**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **項番** | **変数名** | **HCL設定** | **メンバー変数** | **代入順序** | **具体値** |
| 1 | VAR\_hoge | OFF | [0].IP | 入力不要 | 192.168.1.1 |
| 2 | VAR\_hoge | OFF | [0].NAME | 入力不要 | yamamoto |
| 3 | VAR\_hoge | OFF | [1].IP | 入力不要 | 192.168.1.2 |
| 4 | VAR\_hoge | OFF | [1].NAME | 入力不要 | suzuki |
| 5 | VAR\_hoge | OFF | [2].IP | 入力不要 |  |
| 6 | VAR\_hoge | OFF | [2].NAME | 入力不要 |  |

**TerraformのVariablesに登録される値**

|  |  |
| --- | --- |
| **Key** | **Value** |
| VAR\_hoge | [  { “IP” = “192.168.1.1”, “NAME” = “yamamoto” },  { “IP” = “192.168.1.2”, “NAME” = “suzuki” },  ] |

**⑧ 変数値の設定**

変数ネスト管理更新により

メンバー変数「[2].IP」、「[2].NAME」はプルダウンで選択できません。