



IT Automation

CI/CD for IaC 【実習編】

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

Exastro IT Automation Version 1.8
Exastro developer

目次

1. はじめに

- 1.1 [本書について](#)
- 1.2 [作業環境](#)
- 1.3 [シナリオ](#)
- 1.4 [事前準備](#)

2. 実習

- 2.1 [リモートリポジトリの登録](#)
- 2.2 [登録アカウントの登録](#)
- 2.3 [資材紐付の登録](#)
- 2.4 [ドライランで実行\(1回目\)](#)
- 2.5 [Playbookの修正](#)
- 2.6 [ドライランで実行\(2回目\)](#)
- 2.7 [ターゲットサーバへ実行](#)

1. はじめに

1.1 本書について

メインメニュー

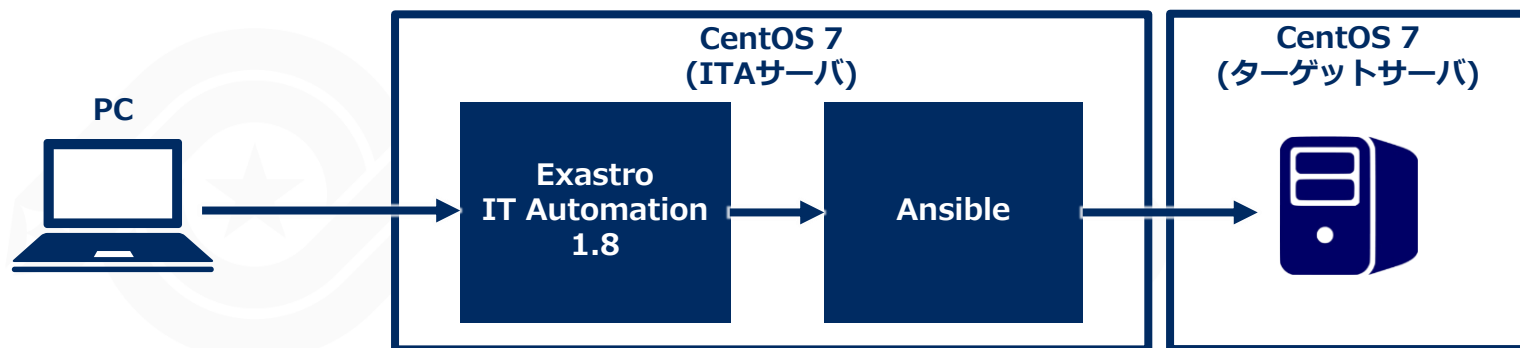
- 本書では、メニューグループの「**CI/CD for IaC**」について、実践形式で学習いただけます。
- なお本書を実施していただく前に「[クイックスタート](#)」の実施が必須となります。



1.2 作業環境

作業環境

- 本書で使用する作業環境は以下の通りです。
- ITAサーバ、ターゲットサーバの他に、GitHubのアカウントをご用意ください。



- Exastro IT Automation 1.8
- CentOS 7(ITAサーバ用)
- CentOS 7(ターゲットマシン用)

1.3 シナリオ

シナリオについて

- 今回のシナリオは以下となります。

① リモートリポジトリの登録

② 登録アカウントの登録

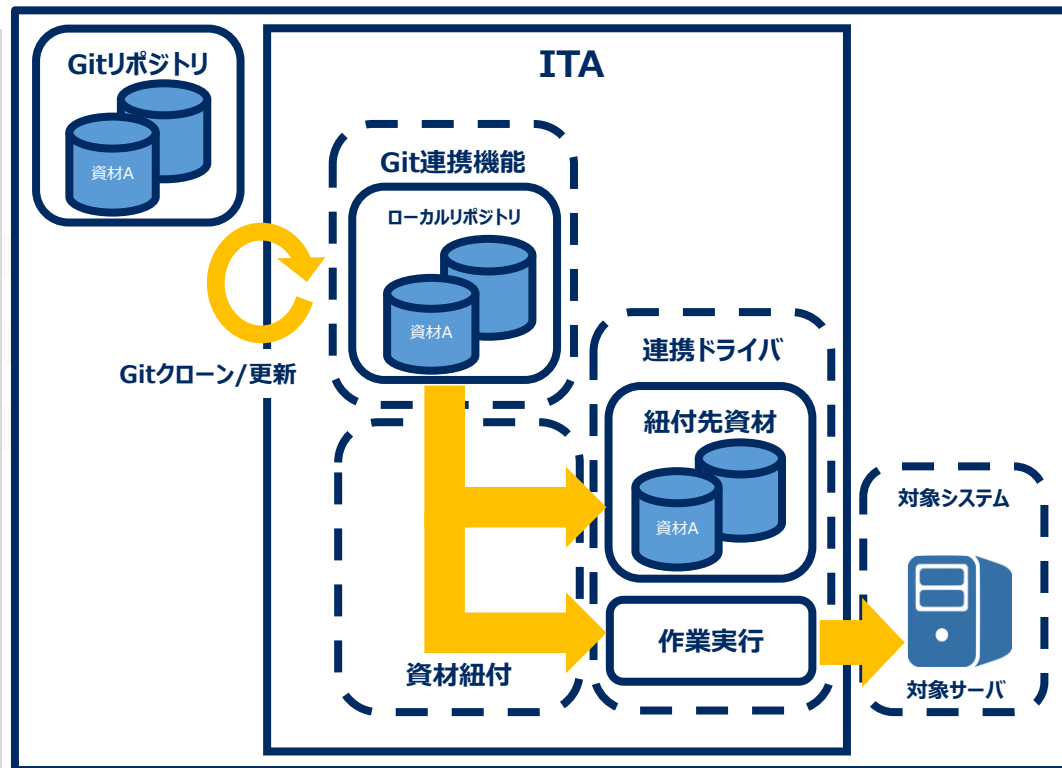
③ 資材紐付の登録

④ ドライランで実行(1回目)

⑤ Playbookの修正

⑥ ドライランで実行(2回目)

⑦ ターゲットサーバへ実行



1.4 事前準備(1/3)

Gitリポジトリの準備

- 今回はGitHubを使用していきます。

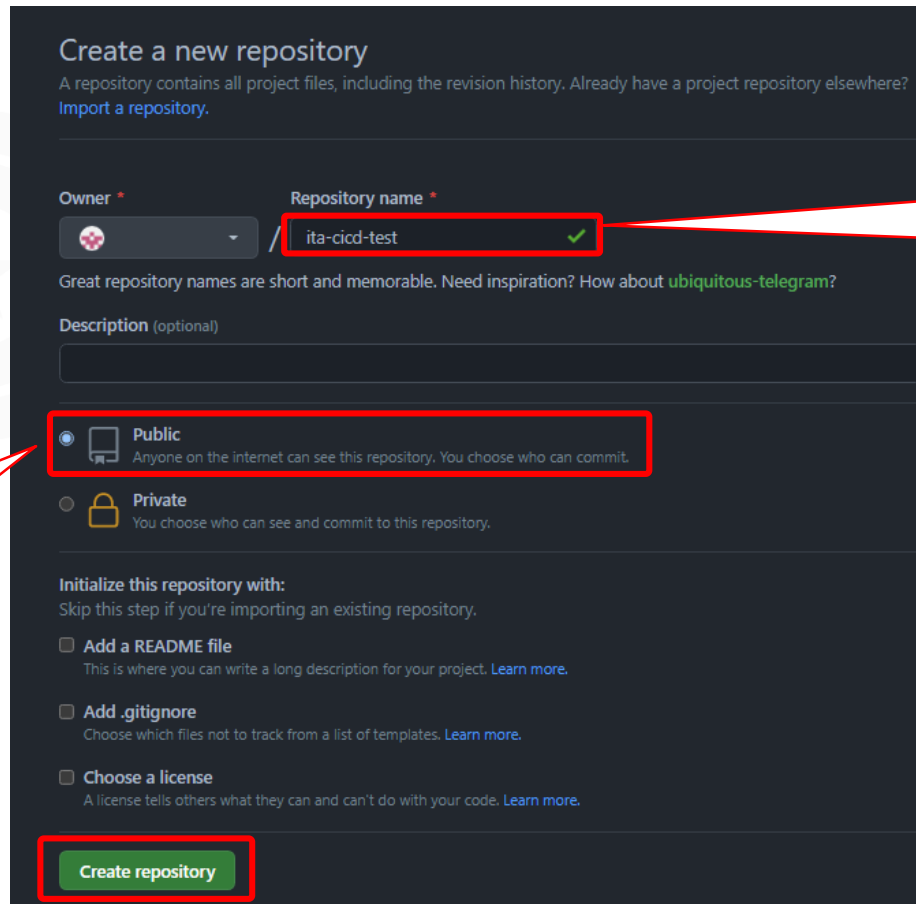

Repository→Newを選択し新規のリポジトリを作成してください。

リポジトリ名を入力しPublicを選択、Create repositoryをクリックし作成して下さい。

Create a new repository


A repository contains all project files, including the revision history. Already have a project repository elsewhere? [Import a repository.](#)


Owner * Repository name *

 ita-cicd-test 

Great repository names are short and memorable. Need inspiration? How about [ubiquitous-telegram?](#)

Description (optional)

☒  **Public**
Anyone on the internet can see this repository. You choose who can commit.

☐  **Private**
You choose who can see and commit to this repository.

Initialize this repository with:

Skip this step if you're importing an existing repository.

☐ **Add a README file**
This is where you can write a long description for your project. [Learn more.](#)

☐ **Add .gitignore**
Choose which files not to track from a list of templates. [Learn more.](#)

☐ **Choose a license**
A license tells others what they can and can't do with your code. [Learn more.](#)

Create repository

今回は
Publicを選択

リポジトリ名
(任意の名前)

1.4 事前準備(2/3)

Playbookの準備

- 今回は以下のPlaybookを使用します。

yum_package_install_check.yml

```
- name: install the latest version of packages
  yum:
    name: "{{ item }}"
    state: latest
  with_items:
    - "{{ VAR_packages }}"

- name: Check yum list
  shell: yum list installed | grep "{{ item }}"
  register: result
  with_items:
    - "{{ VAR_packages }}"
```

Point

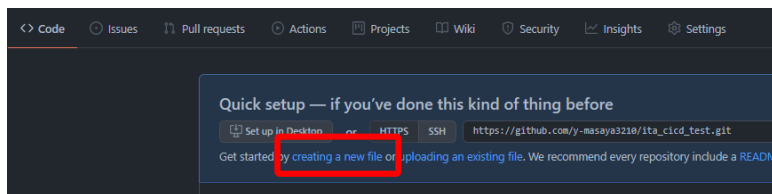
※CI/CDの機能を体感していただくために今回はあえてインデントがずれているものを使用しています。

1.4 事前準備(3/3)

Playbookのアップロード

- GitHubにPlaybookをアップロードします。

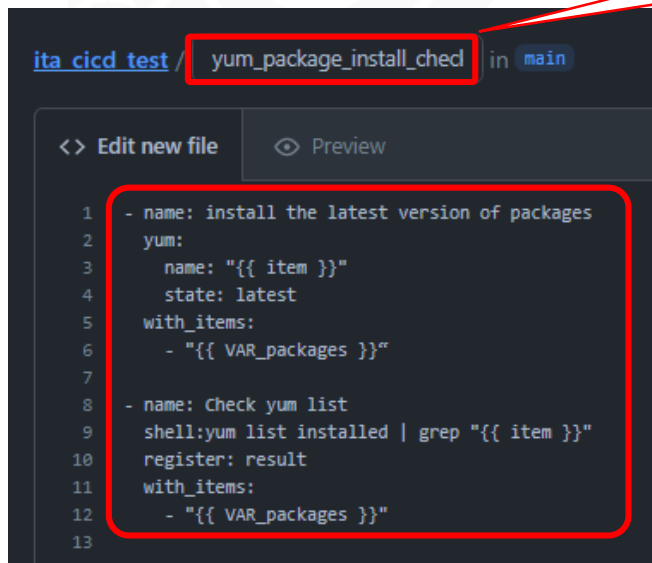
Codeの画面から「creating a new file」をクリック、先ほどのPlaybook名と中身を貼り付けて下にスクロールし「Commit new file」をクリックして下さい。



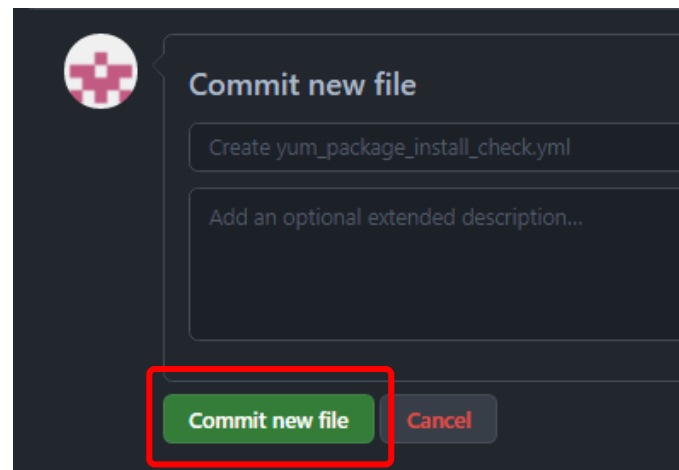
リポジトリ名

yum_package_install_check.yml

中身は[こちら](#)のページを参照。



下へスクロール




2. 実習

2.1 リモートリポジトリの登録

連携するGitリポジトリの情報を登録

- PlaybookをアップロードしたGitHubアカウントの情報を登録します。

「CI/CD for IaC」メニュー→「リモートリポジトリ」をクリック、各項目へ下表のように入力し「登録」をクリックして下さい。

 **CI/CD for IaC**

ようこそ[システム管理者]さん
ログインID [administrator]
[パスワード変更](#) [ログアウト](#)

Menu

メインメニュー

リモートリポジトリ

登録アカウント

資料紐付

説明 ▾開く

表示フィルタ ▾開く

一覧/更新 ▾開く

登録 △閉じる

項番	リモートリポジトリ名*	リモートリポジトリ(URL)*	ブランチ	プロトコル*	Visibilityタイプ	Git アカウント		最終更新日時	最終更新者
						ユーザ	パスワード		
自動入力	<input type="text" value="ita_cicd_test"/>	<input type="text" value="https://github.com"/>	<input type="text"/>	<input type="text" value="https"/>	<input type="text" value="public"/>	<input type="text"/>	<input type="text"/>	自動入力	自動入力

※*は必須項目です。

[戻る](#) [登録](#)

リモートリポジトリ名	リモートリポジトリ(URL)	プロトコル	Visibilityタイプ	リモートリポジトリ同期情報(自動同期)
(事前準備で用意したでリポジトリ名)	(事前準備で用意したリポジトリのURL)	https	public	有効

2.2 登録アカウントの登録

■ 紐付先資材にアクセスするためのアカウント情報の登録

- Exastro IT Automationのアカウントを登録します。

今回はadministratorを使用します。

「登録アカウント」をクリック、各項目へ下表のように入力し「登録」をクリックして下さい。

Menu

メインメニュー

リモートリポジトリ

登録アカウント

資材細付

説明

表示フィルタ

一覧/更新

登録

項番	Exastro IT Automationアカウント		アクセス権		備考	最終更新日時	最終更新者
	ログインID*	ログインPW*	設定	アクセス許可ロール			
自動入力	1:administrator	設定			自動入力	自動入力

※*は必須項目です。

戻る

登録

ログインID

administrator

ログインPW

(任意で設定したPW)

2.3 資材紐付の登録(1/2)

紐付元資材と紐付先資材の紐付を登録

- 紐付元資材と紐付先資材を紐付し、紐付先資材の動作検証を行うためのオペレーションと Movement を登録します。「資材紐付」をクリック、各項目へ下表のように入力して下さい。(次ページへ続く)

Menu

メインメニュー

リモートリポジトリ

登録アカウント

資材紐付

説明

表示フィルタ

一覧/更新

登録

項番	紐付先資材名*	Git リポジトリ (From)		Exastro	最終更新日時	最終更新者
		リモートリポジトリ*	資材パス*	紐付先資材タイプ*		
自動入力	yum_package_install	ita_cicd_test	yum_package_install_check.yml	Ansible-Legacyコンソール/Playbook素材集	自動入力	自動入力

※*は必須項目です。

戻る登録

紐付先資材名	リモートリポジトリ	資材パス	紐付先資材タイプ	実行ログインID	自動同期
yum_package_install	ita_cicd_test	yum_package_install_check.yml	Ansible-Legacy コンソール /Playbook素材集	1:administrator	有効

2.3 資材紐付の登録(2/2)

オペレーションとMovementを登録、ドライランの選択

- 前頁の入力が完了したら右へスクロールし項目へ下表のように入力し、「登録」をクリックして下さい。

Menu

メインメニュー

リモートリポジトリ

登録アカウント

資材紐付

説明

表示フィルタ

一覧/更新

登録

素材同期情報		デリバリ情報			アクセス権			最終更新日時	最終更新者
項番	自動同期	オペレーション	Movement	ドライラン	設定	アクセス許可ロール			
自動入力		オペレーション2	パッケージインストール	●	設定			自動入力	自動入力

戻る 登録

オペレーション	Movement	ドライラン
オペレーション2	パッケージインストール	●

2.4 ドライランで実行確認(1回目)(1/2)

■ ドライランが実行されているか確認

- 資材紐付の登録が完了すると自動的にドライランが実行されます。

「Ansible Legacy」メニュー→「作業管理」をクリック、「フィルタ」をクリックすると実行された作業の一覧が表示されます。対象の作業の「作業状況確認」をクリックしエラーの確認を行います。

Menu

- メインメニュー
- Movement一覧
- Playbook素材集
- Movement-Playbook紐付
- 代入値自動登録設定
- 作業対象ホスト
- 代入値管理
- 作業実行
- 作業状況確認
- 作業管理**

説明 ▽開く

表示フィルタ △閉じる

廃止	作業No.	実行種別	ステータス	実行エンジン	virtualenv	呼	最終更新日時	最終更新者
廃止含まず ▾	<input type="text"/> ~ <input type="text"/> ▼ブルダウン検索	▼ブルダウン検索	▼ブルダウン検索	▼ブルダウン検索	▼ブルダウン検索	▼	<input type="text"/> ~ <input type="text"/>	▼ブルダウン検索

フィルタ **フィルタクリア**

☒ オートフィルタ

一覧 △閉じる

履歴	作業No.	作業状況確認	実行種別	ステータス	実行エンジン	virtualenv	呼出元Symphony	呼出元Conductor	実行コ	最終更新日時	最終更新者
履歴	41	作業状況確認	ドライラン	完了(異常)	Ansible Engine				システ	2021/09/28 17:18:07	legacy作業実行プロシージャ
履歴	40	作業状況確認	ドライラン	完了	Ansible Engine				システ	2021/09/28 17:14:15	legacy作業実行プロシージャ
履歴	39	作業状況確認	ドライラン	想定外エラー	Ansible Engine				システ	2021/09/28 16:56:26	legacy作業実行プロシージャ
履歴	38	作業状況確認	ドライラン	完了(異常)	Ansible Engine				システ	2021/09/28 16:52:19	legacy作業実行プロシージャ
履歴	37	作業状況確認	ドライラン	完了(異常)	Ansible Engine				システ	2021/09/28 16:50:18	legacy作業実行プロシージャ

管理者に連絡

2.4 ドライランで実行確認(1回目)(2/2)

■ ドライランが実行されているか確認

- 進行状況(エラーログ)でエラーを確認することができます。今回はインデントがずれているものを登録したためエラーが発生してしまいました。

Menu

- メインメニュー
- Movement一覧
- Playbook素材集
- Movement-Playbook紐付
- 代入値自動登録設定
- 作業対象ホスト
- 代入値管理
- 作業実行
- 作業状態確認
- 作業管理

進行状況(エラーログ)

フィルタ: ☐ 該当行のみ表示

ERROR! We were unable to read either as JSON nor YAML, these are the errors we got from each:
JSON: Expecting value: line 1 column 1 (char 0)

Syntax Error while loading YAML.
could not find expected '':'

The error appears to be in '/exastro/data_relay_storage/ansible_driver/legacy/ns/0000000041/in/child_playbooks/0000000001-yum_package_
be elsewhere in the file depending on the exact syntax problem.

The offending line appears to be:

```
shell:yum list installed | grep "{{ item }}"
register: result
^ here
```

緊急停止

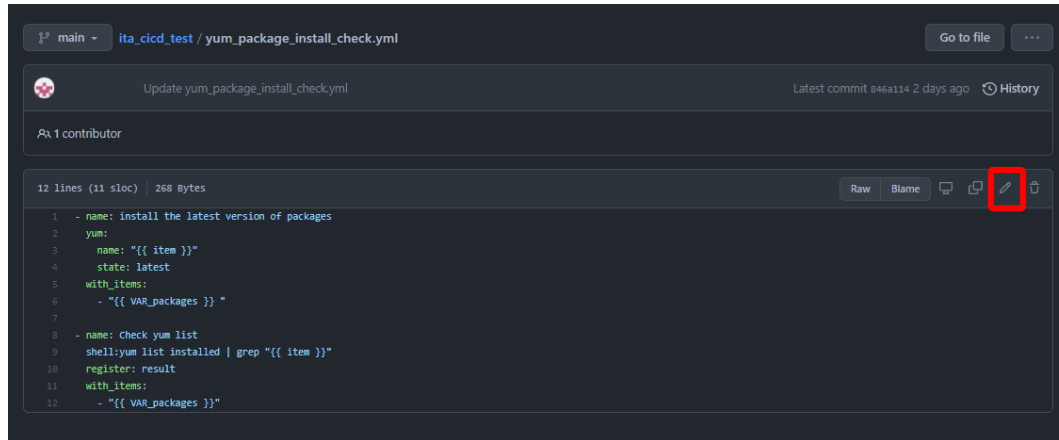
緊急停止

2.5 Playbookの修正

再度GitHubにアクセスしPlaybookを修正

- 先ほどエラーが出てしまった箇所の修正を行っていきます。

再度GitHubにアクセスし編集アイコンをクリック、対象箇所の修正が完了したら「commit changes」をクリックして下さい。



This screenshot shows the GitHub interface for editing a file named 'yum_package_install_check.yml'. The file content is as follows:

```
1 - name: install the latest version of packages
2   yum:
3     name: "{{ item }}"
4     state: latest
5     with_items:
6       - "{{ VAR_packages }}"
7
8 - name: Check yum list
9   shell: yum list installed | grep "{{ item }}"
10  register: result
11  with_items:
12    - "{{ VAR_packages }}"
```

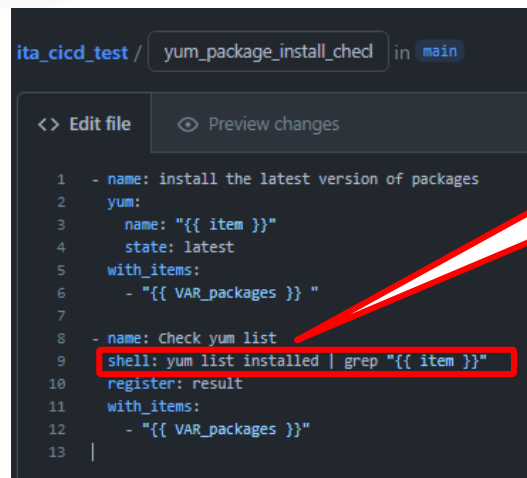
A red box highlights the edit icon (pencil) in the top right corner of the file view.

: とyの間に半角スペースを入れる

```
shell:yum list installed | grep "{{ item }}"
```

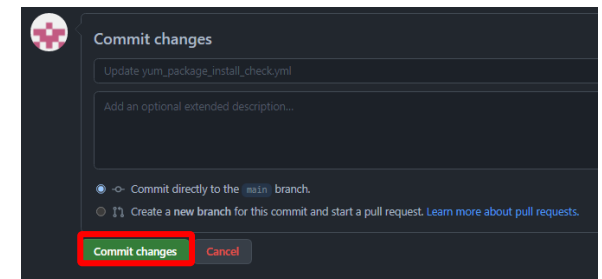


```
shell: yum list installed | grep "{{ item }}"
```



This screenshot shows the 'Edit file' interface in GitHub. The file content is the same as in the previous screenshot. A red box highlights the line `shell: yum list installed | grep "{{ item }}"`, and a red arrow points from the text '下へスクロール' (Scroll down) to this line.

下へスクロール



This screenshot shows the 'Commit changes' dialog box. The 'Commit changes' button is highlighted with a red box.

2.6 ドライランで実行確認(2回目)

■ ドライランが実行されているか確認

- GitHubで更新が完了すると自動的にITAのPlaybookも更新されドライランが実行されます。1回目と同様に「Ansible Legacy」メニュー→「作業管理」をクリック、「フィルタ」をクリックして下さい。1回目は完了の後に(異常)の表示がありましたが、今回は問題なく完了しました。

The screenshot displays the ITA web interface. On the left, the 'Menu' sidebar has '作業管理' (Job Management) selected. The main content area is titled '説明' (Description) and '表示フィルタ' (Display Filter). Below this is a table with columns for '廃止' (Cancel), '作業No.' (Job No.), '実行種別' (Execution Type), 'ステータス' (Status), '実行エンジン' (Execution Engine), 'virtualenv', '呼出元Symphony' (Caller Symphony), '呼出元Conductor' (Caller Conductor), '実行' (Execution), '最終更新日時' (Last Update Time), and '最終更新者' (Last Updated By). The 'フィルタ' (Filter) button is highlighted with a red box. Below the filter buttons, the '一覧' (List) section shows a table of jobs. The first row is highlighted in red, indicating a successful 'Dry Run' (ドライラン) for 'Package Installation' (パッケージインストール) on 2021/10/01.

履歴	作業No.	作業状態確認	実行種別	ステータス	実行エンジン	virtualenv	呼出元Symphony	呼出元Conductor	実行ユーザ	ID	名称	遅延タイム	最終更新日時	最終更新者
履歴	42	作業状態確認	ドライラン	完了	Ansible Engine				システム管理者	1	パッケージインストール		2021/10/01 09:08:46	legacy作業実行プロシージャ
履歴	41	作業状態確認	ドライラン	完了(異常)	Ansible Engine				システム管理者	1	パッケージインストール		2021/09/28 17:18:07	legacy作業実行プロシージャ
履歴	40	作業状態確認	ドライラン	完了	Ansible Engine				システム管理者	1	パッケージインストール		2021/09/28 17:14:15	legacy作業実行プロシージャ
履歴	39	作業状態確認	ドライラン	想定外エラー	Ansible Engine				システム管理者	1	パッケージインストール		2021/09/28 16:56:26	legacy作業実行プロシージャ
履歴	38	作業状態確認	ドライラン	完了(異常)	Ansible Engine				システム管理者	1	パッケージインストール		2021/09/28 16:52:19	legacy作業実行プロシージャ
履歴	37	作業状態確認	ドライラン	完了(異常)	Ansible Engine				システム管理者	1	パッケージインストール		2021/09/28 16:50:18	legacy作業実行プロシージャ
履歴	36	作業状態確認	ドライラン	完了(異常)	Ansible Engine				システム管理者	1	パッケージインストール		2021/09/28 16:45:11	legacy作業実行プロシージャ
履歴	35	作業状態確認	ドライラン	想定外エラー	Ansible Engine				システム管理者	1	パッケージインストール		2021/09/28 16:43:05	legacy作業実行プロシージャ

2.7 ターゲットサーバへ実行(1/2)

作業実行から実際にターゲットサーバへ実行

- 先ほどまではドライランで実行しPlaybookの記載が問題ないかチェックをしていましたが修正が完了問題なく動作したのでいよいよ実際のターゲットサーバに反映させていきます。「Ansible Legacy」メニュー→「作業実行」をクリック、実行するMovementとオペレーションを選択します。(次ページへ進む)

Menu

メインメニュー

Movement一覧

Playbook素材集

Movement-Playbook紐付

代入値自動登録設定

作業対象ホスト

代入値管理

作業実行

作業状態確認

作業管理

説明

▼開く

スケジュールリング

△閉じる

予約日時を指定する場合は、日時フォーマット(YYYY/MM/DD HH:II)で入力して下さい。ブランクの場合は即時実行となります

予約日時

Movement[フィルタ]

▼開く

Movement[一覧]

△閉じる

選択	MovementID	Movement名	オクストレータ	遅延タイマー	Ansible利用情報				アクセス権	備考	最終更新日時	最終更新者
					ホスト指定形式	WinRM接続	ヘッダーセクション	オプションパラメータ	アクセス許可ロール			
<input checked="" type="radio"/>	1	パッケージインストール	Ansible Legacy		IP						2021/08/05 19:45:31	システム管理者

フィルタ結果件数: 1

オペレーション[フィルタ]

▼開く

オペレーション[一覧]

△閉じる

選択	No.	オペレーションID	オペレーション名	実施予定日時	最終実行日時	アクセス権	備考	最終更新日時	最終更新者
						アクセス許可ロール			
<input type="radio"/>	1	1	オペレーション1	2021/05/02 12:00	2021/09/16 09:45			2021/09/16 09:45:16	legacy作業実行プロシージャ
<input checked="" type="radio"/>	2	2	オペレーション2	2021/05/29 18:00	2021/10/01 09:07			2021/10/01 09:07:57	legacy作業実行プロシージャ

管理者に連絡

2.7 ターゲットサーバへ実行(2/2)

作業実行から実際にターゲットサーバへ実行

- 下へスクロールし実行をクリック、実行しますかのポップアップが出るので「OK」をクリックして下さい。実行されステータスが完了の表示になったら無事に反映完了です。

オペレーション[フィルタ]

オペレーション[一覧]

選択	No.	オペレーションID	オペレーション名	実施予定日時	最終実行日時	アクセス権 アクセス許可ロール	備考	最終更新日時	最終更新者
<input type="radio"/>	1	1	オペレーション1	2021/05/02 12:00	2021/09/16 09:45			2021/09/16 09:45:16	
<input checked="" type="radio"/>	2	2	オペレーション2	2021/05/29 18:00	2021/10/01 09:07			2021/10/01 09:07:57	

フィルタ結果件数: 2

MovementID 1
Movement名 パッケージインストール

ドライラン

実行

項目		値
作業No.		43
実行種別		通常
ステータス		完了
実行エンジン		Ansible Engine
呼出元Symphony		
呼出元Conductor		
実行ユーザ		システム管理者
Movement	ID	1
	名称	パッケージインストール
	遅延タイム(分)	
	Ansible利用情報	ホスト指定形式 IP WinRM接続
オペレーション	No.	2
	名称	オペレーション2
	ID	2
作業対象ホスト		確認
代入値		確認
入力データ	投入データ	InputData_0000000043.zip
出力データ	結果データ	ResultData_0000000043.zip
作業状況	予約日時	
	開始日時	2021/10/01 11:02:53
	終了日時	2021/10/01 11:03:46



Exastro