

metabaseマニュアル トライアル用

ver.2.2
connectome.design inc.

改定履歴

2022.01	ver.1.0	初版作成。
2022.02	ver.1.1	機能追加に伴う加筆。
2022.03	ver.2.0	フォーマット変更と機能改修に伴う加筆。
2022.03	ver.2.0.1	自動翻訳に合わせ一部文言修正。
2022.03	ver.2.1	ファイル名とタイトルを変更。
2022.05	ver.2.2	サンプルDEKを”boston”から”california”に変更。

metabase利用開始について

- サービス形態。
 - AWS上に構築されたSaaS。
 - サービス案内webサイト: <https://metabase.jp>
- 推奨動作環境。
 - Google Chrome
- ユーザ登録は、ラボ単位での招待制。
- ユーザ登録には、メールアドレスが必要。
- ユーザ登録の概略手順は以下の通り。
 - 利用者は事前に利用手続き(招待をもらうための申請)を行う。
 - アクセス権限管理のため、ラボ管理者が招待操作を行う。
 - 招待されたユーザがサインアップする。
 - ユーザがサインアップした後、ラボ管理者がプロジェクトにメンバーとして追加する。
- アクセスURL: <https://trial.metabase.online>

用語集

- **DEK**: Distributable Expandable Knowledge / “流通可能かつ拡張可能な知”。metabaseで扱う“ひと塊の情報”(プログラム、データ、文書等)を表す造語。
※読み方は、“デッキ”。
- **ラボ**: チームのひとまとまりの単位。ユーザはラボ単位で招待される。
- **プロジェクト**: ラボ内での案件のまとまり。ユーザはプロジェクト単位でアクセス権を付与される。DEKはプロジェクト単位で保持される。
- **タスク**: DEK(プログラム+データ)をVMタイプ上で動かすための設定情報。
- **VMタイプ**: タスクの実行環境の定義。

サインアップ(ユーザ登録) / ログインの手順

1. サインアップ(ユーザ登録)から始める場合。
 - a. 招待メールが届いたら、メール内のリンクからサインアップへ進む。
(メールの件名: [metabase] ラボへ招待されました / You are invited to the Lab)
 - b. しばらくすると登録確認のメールが届くので、メール内のリンクボタン [VERIFY YOUR ACCOUNT] をクリックし登録完了。
(メールの件名: "Verify your email")
 - c. 招待されたラボへの登録も同時に完了する。

2. サインアップ済みユーザの場合。
 - a. アクセスURL(最初のページに記載)にアクセスし、IDとPWを入力してログイン。

ラボへの参加手順

1. サインアップ(ユーザ登録)から始める場合。
 - a. 前ページの手順でサインアップが完了すると、招待されたラボへの参加も完了。
2. サインアップ済みユーザの場合。
 - a. 招待メールが届いたら、メール内のリンクからログインし、画面右上の通知アイコンから内容を確認する。
(メールの件名:[metabase] ラボへ招待されました/You are invited to the Lab)
 - b. 【要対応】と記載されているメッセージをクリックし、ラボへの招待を受ける。

metabaseチュートリアル[基本操作手順]

- metabaseへブラウザでアクセスし、ログインした状態で行う。
(※アクセスURLは、最初のページを参照)
- 操作がない状態で一定時間経過すると自動的にログアウトする。

プログラムを動作させるための概略手順

1. ラボに参加し、ログインする。
2. プロジェクトを作成する。
3. MarketplaceからDEKを入手する。
4. DEKとVMタイプを選んでタスクを作成する。
5. タスクを起動する。
6. JupyterLabを開いて、プログラムを動かす。
7. タスクを停止させる。

※以下に詳細な手順を示す。

プロジェクトの作成

1. グローバルメニュー[Lab] → ラボ一覧から選択する。
 - a. 参加しているラボのみが表示される。
2. [+プロジェクト作成]から、プロジェクトを新規作成する。
 - a. プロジェクト名: 自由。自分が作ったものであることがわかる名前が望ましい。
 - b. プロジェクト説明: 任意。
 - c. メンバー: 自分が管理者として初期設定されているのでそのままよい。
3. [登録]。

MarketplaceからプログラムDEKを入手し、タスクに読み込んで実行する例

1. DEKの入手。
 - a. グローバルメニュー(画面最上部の全体メニュー)から[Marketplace]へ移動する。
 - b. プログラムDEKから、一つを選択し詳細画面を表示する。
(例:"california_scikit-learn")
 - c. [入手]→入手先設定が開く。
 - i. 購入先ラボ:自分で作成したプロジェクトのラボを選択する。
 - ii. 購入先プロジェクト:自分で作成したプロジェクトを選択する。
 - iii. 保存DEK名:デフォルトのまま。
 - d. [入手]→[購入先へ移動]→プロジェクト内にDEKを入手できたことが確認できる。
2. タスクの作成。
 - a. [タスク]タブへ移動。
 - b. [+タスク作成]から、タスクを新規作成する。
 - i. タスク名:自由。タスクの機能がわかる名前が望ましい。
 - ii. タスク説明:任意。
 - iii. DEKを選択する。
(例:"プログラムDEK:california_scikit-learn")
 - iv. VMタイプ:【推奨】と記載のあるものを選択する。
 - v. 最大稼働時間:デフォルト(3600)のまま。
 - c. [登録]。
3. タスクの起動。
 - a. [タスク]タブへ移動。
 - b. タスク名一覧から先ほど作成したタスクを選択し、[操作]ボタンから、[起動]を選択する。
 - c. 起動確認画面が開くので内容確認し、[起動]。
 - d. ステータスが「**起動中**」になるのでそのまま待機する。
(AWSの利用状況により、起動には5分程度かかる場合がある。)
 - e. ステータスが「**稼働中**」になったら起動完了。
 - f. [操作]ボタンから[開く]を選択する。
 - g. ブラウザの別タブで、JupyterLabが開く。
4. プログラムの実行。
 - a. JupyterLabが開くと、読み込まれたDEKが[source]フォルダに配置されていることが確認できる。
 - b. [source]フォルダ内のzipファイルを展開する。
 - i. 後述「JupyterLabでのmetabase用コンテキストメニューの実行方法」を参照。

- c. 展開されたフォルダ内に、“requirements.txt”が存在する場合は、必要に応じて、ライブラリの追加インストールを行う。
 - i. 後述「JupyterLabでのmetabase用コンテキストメニューの実行方法」を参照。
- d. 展開したフォルダを[work]フォルダへ移動させ、[work]フォルダにアクセスする。
 - i. 後述「JupyterLabでのmetabase用コンテキストメニューの実行方法」を参照。
 - ii. 実行結果等の保存が不要な場合は、移動せずに[source]フォルダからも実行可能。
- e. プログラムフォルダに移動する。
(例:“california_scikit-learn”)
- f. プログラムファイルを開く。
(例:“california_scikit-learn.ipynb”)
 - i. Kernel選択の画面が表示されたら、“Python 3”を選ぶ。
 - ii. Jupyter Notebook上にソースコードが開く。
- g. Runボタンで一行ずつ実行する。
 - i. 実行途中でimportエラー(ModuleNotFoundError)が発生する場合は、ライブラリを追加インストールする。
 - ii. 後述「JupyterLabでのmetabase用コンテキストメニューの実行方法」を参照。
 - iii. ライブラリが追加インストールされたらステップを再実行する。
- h. 実行結果ファイルが出力される場合は、[work]フォルダ内に保存される。

タスクを停止する手順

1. 実行が終わったら、Fileメニューから [Close and Shutdown] を選んでソースコードを閉じる。
 - a. その他の操作で閉じると、プロセスが実行中のままとなりメモリが開放されない。
 - b. 実行中のプロセスは、画面左側のタブから確認し、不要なKernelが残っていたら、Shutdownする。
[Running Terminals and Kernels]
2. metabaseのタスク画面に戻り、[操作]ボタンから[停止]を選択する。
(Shutdownを行わず、ソースコードを開いたまま停止してもよい。)
3. [work]フォルダ配下のファイルは、タスク停止後も永続化され、次回タスク起動時に復元される。
(ただし、中身が空のフォルダは保存されない。)
4. 画面上でソースコードを変更し、[work]フォルダに保存することも可能。
5. タスク稼働時間の初期値は3600秒。
 - a. タスク稼働時間はJupyterLabの右上[+1 hour]ボタンで1時間ずつ延長可能。
6. 動作確認が終わったら、忘れずに停止すること。
 - a. 停止しない場合、稼働設定時間に対する課金が発生する。

手持ちのプログラムをDEKとして登録し動かす例

- ローカルPC上で、登録したいプログラム一式を一つのフォルダにまとめる。
 - プログラム格納用フォルダを作り(例:"helloworld")、その中にプログラムファイルを入れる(例:"helloworld.ipynb")。
 - 追加インストールが必要なライブラリがある場合は、"requirements.txt"を作成し、フォルダの最上位に配置しておく。
- そのフォルダを、zip圧縮する。
(例:"helloworld.zip")
- metabaseにログインし、自分のプロジェクトを開く。
- [DEK]タブを開き、[+DEK作成]を選択。
- DEK名など必要事項を設定し、作ったzipファイルをアップロードして[登録]。
 - 出品時デフォルト設定は、[価格:無料]、[公開設定:公開]を選択しておけばよい。
- [DEK]タブを開くと、作成したDEKが一覧に表示される。
- [タスク]タブへ移動し、[+タスク作成]から、タスクを新規作成する。
 - タスク名:自由。「タスクの機能がわかる名前」が望ましい。
 - タスク説明:任意。
 - DEK:先ほど登録したDEKを選択する。
 - VMタイプ:【推奨】と記載のあるものを選択する。
 - 最大稼働時間:デフォルト(3600)のまま。
- [登録]。
- 以降の操作は、前述の「タスクの起動」と同じ。

ゼロからmetabase上でプログラムを作成し動かす例

1. プロジェクトを選択する もしくは新たに作成する。
2. 利用するデータがあれば、データDEKとして登録する。
 - a. プログラムDEKと同様、一つのフォルダにまとめてzip圧縮してDEK登録する。
 - b. MarketplaceからデータDEKを入手することも可能。
3. タスクを作成する。
 - a. 必要であれば、データDEKを指定する。
4. タスクを起動し、JupyterLabを開く。
 - a. データDEKは[source]フォルダに配置されるのでzipファイルを展開し、[work]フォルダへ移動する。
5. [work]フォルダ内にフォルダを新規作成し、JupyterNotebookを開いてプログラムを作成する。
 - a. タスク停止時に作成したファイルが永続化されるよう、[work]フォルダを使用する。
 - b. プログラムを保存するためのフォルダを一つ作り、その配下にプログラムを配置する。
 - c. 作成したプログラムをDEKとして登録したい場合は、アーカイブしてダウンロードし、別途、DEK登録を行う。
 - i. 後述「JupyterLabでのmetabase用コンテキストメニューの実行方法」を参照。

JupyterLabでのmetabase用コンテキストメニューの実行方法

- JupyterLab内での操作を容易にするため、コンテキストメニュー（右クリックメニュー）にコマンドが用意されている。

1. **Unzip**: zip展開。
2. **Install Dependencies**: "requirements.txt" からのライブラリインストール。
3. **Move To Work**: [work]フォルダへの移動。
4. **Archive For DEK**: プログラムフォルダからDEK用zipファイル作成。

※以下に詳細な使用方法を示す。

1. **Unzip**: zipファイルを展開する。
 - a. JupyterLabの画面左側のフォルダ一覧から、[source]フォルダを開く。
 - b. DEKのzipファイルが配置されているので選択し、コンテキストメニューから [Unzip] を実行する。
(例: "dek-51f82e15****.zip")
 - c. 同じフォルダ内に、DEKのフォルダが展開されたことが確認できる。
2. **Install Dependencies**: "requirements.txt" を読み込んでライブラリを追加インストールする。
 - a. JupyterLabの画面左側のファイル一覧から "requirements.txt" を探す。
 - b. "requirements.txt" を選択し、コンテキストメニューから [Install Dependencies] を実行する。
(※ファイルは選択するだけ。開かない)
 - c. ライブラリがインポートされる。
※このコマンドは、タスクを起動するたびに実行する必要あり。
3. **Move To Work**: [work]フォルダへ移動させる。
 - a. Unzipののち展開したフォルダを選択し、コンテキストメニューから [Move To Work] を実行する。
(※フォルダは選択するだけ。開かない)
 - b. 指定したフォルダが、[work]フォルダへ移動されたことが確認できる。
4. **Archive For DEK**: プログラムフォルダからDEK用zipファイルを作成する。
 - a. JupyterLabの画面左側、フォルダ一覧から、作成したプログラムが配置されたフォルダを選択し、コンテキストメニューから [Archive For DEK] を実行する。
(※フォルダは選択するだけ。開かない)
 - b. 選択したフォルダからアーカイブzipファイルが作成されたことを確認できる。
 - c. 作成されたzipファイルはダウンロードできる。

- コマンドを実行するとTerminal画面が開き、処理が完了すると、“Complete! Please press Enter to exit the terminal.”が表示されるので、EnterキーでTerminal画面を閉じる。
- Terminal画面でメッセージ(既存ファイルへの上書き許可など)が表示された場合は、内容を確認し、指示を入力すること。
- “requirements.txt”の有無に関係なくライブラリを追加インストールしたい場合は、次のどちらかに行う。
 - Jupyter Notebook上で、最上部に“!pip install”のステップを追加し実行する。
 - Terminalを開いて、“pip install”コマンドを実行する。
- どちらの場合も、[source/work]フォルダどちらにて実行してもよいが、タスクを起動するたびに実行する必要あり。
- ダウンロードしたアーカイブzipファイル内には、動作環境に含まれるライブラリがすべて書き出された“requirements.txt”が含まれているので、ローカル環境でzipファイルを展開し、フォルダ内の“requirements.txt”の内容を、“自分で追加した必要最低限のライブラリのみ”に整理する。そのフォルダを再度zip圧縮したファイルは、新たなDEKとして登録できる。