

PASSWORD MANAGER > 開発者ツール

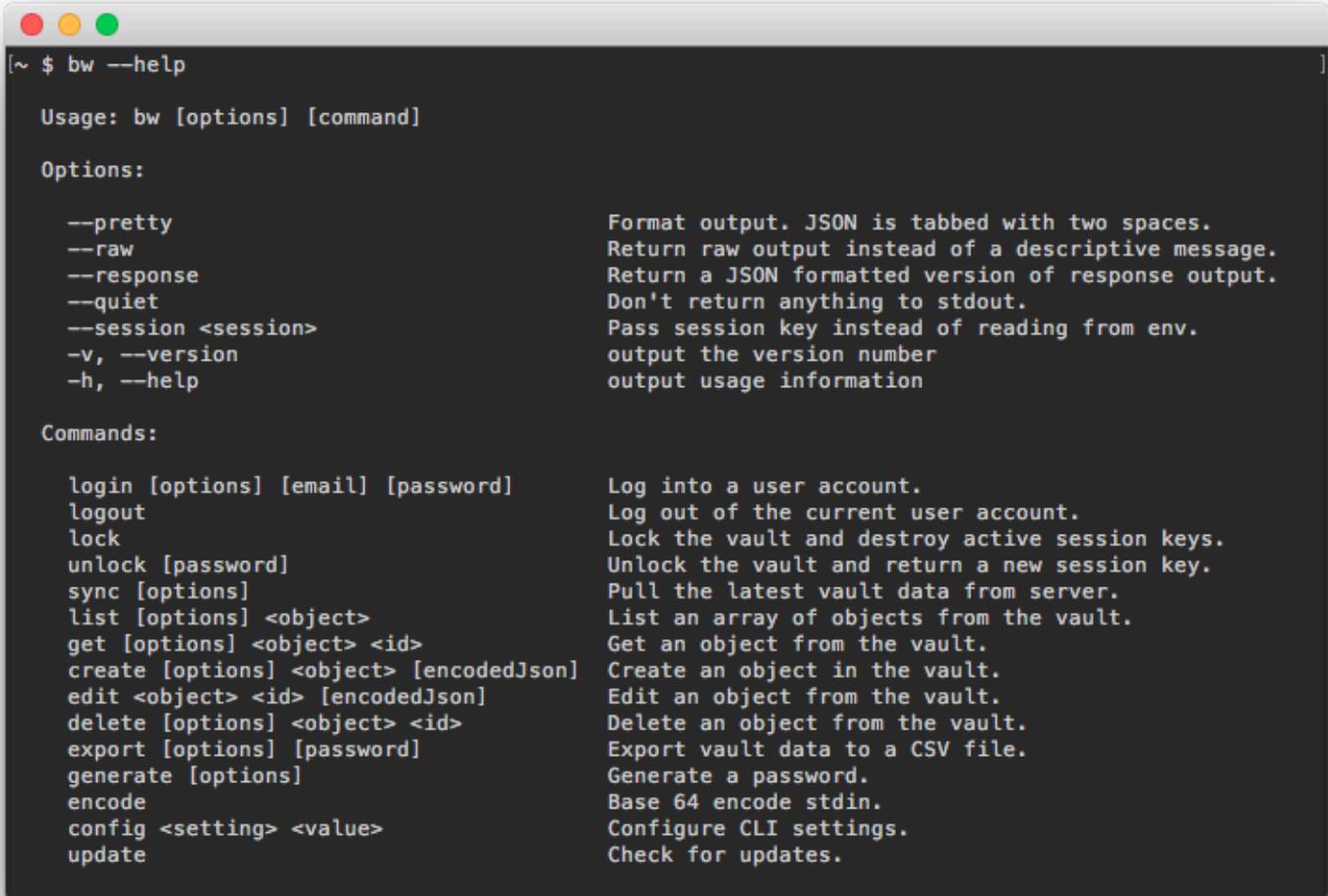
パスワードマネージャー CLI

ヘルプセンターで表示:

<https://bitwarden.com/help/cli/>

パスワードマネージャー CLI

Bitwardenのコマンドラインインターフェイス (CLI) は、保管庫へのアクセスと管理のための強力な完全な機能を備えたツールです。他のBitwardenクライアントアプリケーション (デスクトップ、ブラウザ拡張機能など) で見つけることができるほとんどの機能は、CLIから利用できます。



```
[~ $ bw --help]

Usage: bw [options] [command]

Options:

  --pretty          Format output. JSON is tabbed with two spaces.
  --raw            Return raw output instead of a descriptive message.
  --response       Return a JSON formatted version of response output.
  --quiet          Don't return anything to stdout.
  --session <session> Pass session key instead of reading from env.
  -v, --version    output the version number
  -h, --help      output usage information

Commands:

  login [options] [email] [password]  Log into a user account.
  logout                               Log out of the current user account.
  lock                                 Lock the vault and destroy active session keys.
  unlock [password]                   Unlock the vault and return a new session key.
  sync [options]                       Pull the latest vault data from server.
  list [options] <object>             List an array of objects from the vault.
  get [options] <object> <id>         Get an object from the vault.
  create [options] <object> [encodedJson] Create an object in the vault.
  edit <object> <id> [encodedJson]    Edit an object from the vault.
  delete [options] <object> <id>     Delete an object from the vault.
  export [options] [password]         Export vault data to a CSV file.
  generate [options]                  Generate a password.
  encode                               Base 64 encode stdin.
  config <setting> <value>           Configure CLI settings.
  update                              Check for updates.
```

Bitwarden CLI

Bitwarden CLIは自己文書化されています。コマンドラインから、以下を使用して利用可能なコマンドについて学びます:

Bash

```
bw --help
```

または、利用可能なオプションと例を見るために、任意の**bw**コマンドにオプションとして**--help** を渡してください。

Bash

```
bw list --help
```

```
bw move --help
```

ほとんどの情報は`--help`を使用してアクセスできますが、この記事ではそのすべての情報を再現し、一部のトピックについてはより詳しく説明しています。

ダウンロードしてインストールしてください

CLIはWindows、macOS、およびLinuxディストリビューションでクロスプラットフォームとして使用することができます。Bitwarden CLIをダウンロードしてインストールするには：

Note

arm64デバイスの場合、`npm`を使用してCLIをインストールします。

⇒ネイティブ実行可能ファイル

各プラットフォーム用にネイティブにパッケージ化されたCLIのバージョンが利用可能で、依存関係はありません。これらのリンクのいずれかを使用してダウンロードしてください：

- [Windows x64](#)
- [macOS x64](#)
- [Linux x64](#)

ダウンロードしたネイティブ実行可能ファイルを使用する場合、メモしてください。実行可能ファイルをあなたのPATHに追加するか、ファイルがダウンロードされたディレクトリからコマンドを実行する必要があります。

Tip

LinuxとUNIXシステムでは、**権限が拒否されました**というメッセージが表示されることがあります。もしそうなら、次のコマンドを実行して権限を付与してください：

Bash

```
chmod +x </path/to/executable>
```

⇒NPM

あなたのシステムにNode.jsがインストールされている場合、NPMを使用してCLIをインストールできます。NPMでのインストールは、インストールを最新の状態に保つ最も簡単な方法であり、すでにNPMに慣れている人々にとっては**優先される方法**であるべきです：

Bash

```
npm install -g @bitwarden/cli
```

npmjs.orgでパッケージを表示してください。

📌 Note

Bitwarden CLIをLinuxシステムにnpmを使用してインストールするには、最初にbuild-essential依存関係（または配布版に相当するもの）をインストールする必要があるかもしれません。例えば：

Plain Text

```
apt install build-essential
```

⇒チョコレート風味

Chocolateyでインストールするには：

Bash

```
choco install bitwarden-cli
```

パッケージをcommunity.chocolatey.orgで表示します。

⇒スナップ

スナップでインストールするには：

Bash

```
sudo snap install bw
```

snapcraft.ioでパッケージを表示します。

ログイン

ログインする前に、CLIが正しいサーバー（例えば、EUクラウドまたは自己ホスト型）にconfigコマンドを使用して接続されていることを確認してください（[詳細を学ぶ](#)）。ログインコマンドを使用してBitwarden CLIにログインするための3つの方法があり、それぞれが異なる状況に適しています。次のオプションを見直して、どの方法を使用するかを決定してください：

- メールアドレスとマスターパスワードを使用する
- APIキーを使用する
- SSOを使用して

どのオプションを使用するにせよ、終了するときには必ず**bw ロック**コマンドまたは**bw ログアウト**コマンドを使用してください。

💡 Tip

メールアドレスとマスターパスワードを使用してログインすると、マスターパスワードが使用されるため、**ログイン**と**ロック解除**のコマンドと一緒に連ねて、あなたのIDを認証し、保管庫を同時に復号化することができます。**APIキー**または**SSO**を使用する場合、保管庫のデータを直接扱う場合は、**ログイン**コマンドに明示的な**bw ロック解除**を続ける必要があります。

これは、保管庫のデータを復号化するために必要なキーの源がマスターパスワードであるからです。ただし、保管庫が復号化されていなくても実行できるコマンドがいくつかあります。それらには、**config**、**encode**、**生成**、**更新**、および**status**が含まれます。

メールアドレスとパスワードを使用する

メールアドレスとパスワードでログインすることは、**インタラクティブなセッションに推奨されます**。

メールアドレスとパスワードでログインするには：

Bash

```
bw login
```

これにより、あなたの**メールアドレス**、**マスターパスワード**、そして（有効化されている場合）**二段階ログインコード**のプロンプトが開始されます。CLIは現在、**認証器**、**メールアドレス**、または**YubiKey**を介した二段階ログインをサポートしています。

これらの要素を次の例のように一つのコマンドにまとめることができますが、セキュリティ上の理由から推奨されません：

Bash

```
bw login [email] [password] --method <method> --code <code>
```

[列挙型](#)を参照してください。二段階ログインの値についてです。

💡 Tip

追加の認証を求められるか、**あなたの認証リクエストはボットから来ているようです**。エラーが出ますか？
認証チャレンジに答えるために、あなたのAPIキー**client_secret**を使用してください。[もっと学ぶ](#)

APIキーを使用する

自動化されたワークフローや外部アプリケーションへのアクセスを提供するため、またはあなたのアカウントがCLI（FIDO2またはDuo）でサポートされていない2FA方法を使用している場合、**個人APIキー**でログインすることをお勧めします。APIキーでログインするには：

Bash

```
bw login --apikey
```

これにより、あなたの個人的な `client_id` と `client_secret` のプロンプトが開始されます。
これらの値を使用してセッションが認証されると、`ロック解除` コマンドを使用できます。もっと学ぶ

💡 Tip

組織で SSO が必要な場合でも、`--apikey` を使用して CLI にログインできます。

APIキー環境変数を使用する

Bitwarden CLI を使用して自動化された作業が行われているシナリオでは、認証時の手動介入の必要性を防ぐために、環境変数を保存することができます。

環境変数名	必要な値
BW_CLIENTID	クライアントID
BW_CLIENTSECRET	クライアントシークレット

SSOを使用して

組織が SSO 認証を必要とする場合、SSO でログインすることをお勧めします。SSO でログインするには：

Bash

```
bw login --sso
```

これにより、あなたのウェブブラウザで SSO 認証フローが開始されます。あなたのセッションが認証されたら、`ロック解除` コマンドを使用できます。もっと学ぶ

💡 Tip

組織で SSO が必要な場合は、代わりに `--apikey` を使用して CLI にログインすることもできます。

複数のアカウントにログインしてください

他の Bitwarden アプリでアカウント切り替えを使用するように、CLI は `BITWARDENCLI_APPDATA_DIR` 環境変数を使用して、同時に複数のアカウントにログインする能力があります。この環境変数は通常 `bw` 設定ファイルの場所、通常は `data.json` という名前のファイルを指しています。例えば、2つの別々の設定のために、`.bashrc` プロファイルでエイリアスを設定することができます。

Bash

```
alias bw-personal="BITWARDENCLI_APPDATA_DIR=~/.config/Bitwarden\ CLI\ Personal /path/to/bw $@"  
alias bw-work="BITWARDENCLI_APPDATA_DIR=~/.config/Bitwarden\ CLI\ Work /path/to/bw $@"
```

この例を使用すると、まず `source /path/to/.bashrc` を実行し、次に `bw-personal ログイン` と `bw-work ログイン` を実行することで、二つのアカウントにログインすることができます。

ロック解除

API キーまたはSSOを使用してログインするには、ポールド データを直接操作する場合は、`ログイン` コマンドの後に明示的に `bw ロック` を解除する必要があります。

あなたの保管庫をロック解除すると、**セッションキー**が生成され、それはあなたの保管庫内のデータと対話するために使用される復号化キーとして機能します。セッションキーは使用しなければなりません保管庫データに触れる任意のコマンドを実行するために（例えば、`リスト`、`取得`、`編集`）。セッションキーは、`bw ロック` または `bw ログアウト` を使用して無効化するまで有効ですが、新しいターミナルウィンドウを開くと持続しません。いつでも次のように新しいセッションキーを生成してください：

Bash

```
bw unlock
```

終了するときは、常に `bw ロック` コマンドを使用してセッションを終了してください。

ロック解除オプション

あなたは `--passwordenv` または `--passwordfile` のオプションを `bw ロック解除` と一緒に使用して、マスターパスワードを手動で入力する代わりに取得することができます。例えば：

1. 次のものは、環境変数 `BW_PASSWORD` を探します。 `BW_PASSWORD` が空でなく、正しい値を持っている場合、CLIは成功裏にロック解除し、セッションキーを返します。

Bash

```
bw unlock --passwordenv BW_PASSWORD
```

2. 次の操作は、ファイル `~/Users/Me/Documents/mp.txt`（その最初の行にはマスターパスワードが必要です）を探します。ファイルが空でなく、正しい値を持っている場合、CLIは成功してロック解除し、セッションキーを返します。

Bash

```
bw unlock --passwordfile ~/Users/Me/Documents/mp.txt
```

⚠ Warning

`--passwordfile` オプションを使用する場合は、パスワードファイルを保護し、`bw ロック解除` を実行する必要があるユーザーだけにアクセスをロックし、そのユーザーに対してのみ読み取り専用のアクセスを提供してください。

セッションキーを使用する

あなたが**bw ログイン**を使用してメールアドレスとパスワードで保管庫をロック解除するか、または**bw ロック解除**を使用すると、CLIは**エクスポート BW_セッション** (Bash) と**env:BW_セッション** (PowerShell) コマンドを返し、セッションキーが含まれます。関連するエントリーをコピーして貼り付け、必要な環境変数を保存してください。

BW_SESSION環境変数が設定されていると、**bw**コマンドはその変数を参照し、クリーンに実行することができます。例えば：

Bash

```
export BW_SESSION="5PBYGU+5yt3RHcCjoeJKx/wByU34vokGRZjXpSH7Ylo8w=="
```

```
bw list items
```

あるいは、環境変数を設定しない場合、各**bw**コマンドにオプションとしてセッションキーを渡すことができます。

Bash

```
bw list items --session "5PBYGU+5yt3RHcCjoeJKx/wByU34vokGRZjXpSH7Ylo8w=="
```

💡 Tip

終了するときは、常に**bw ロック**または**bw ログアウト**のコマンドを使用してセッションを終了してください。これはアクティブなセッションキーを無効にします。

コアコマンド

作成する

「**create**」コマンドは、あなたの保管庫に新しいオブジェクト（**アイテム**、**添付ファイル**など）を作成します。

Bash

```
bw create (item|attachment|folder|org-collection) <encodedJson> [options]
```

「**作成**」コマンドはエンコードされたJSONを取ります。オブジェクトを作成するための典型的なワークフローは、次のようになるかもしれません：

1. オブジェクトのタイプに適したJSONテンプレートを出力するには、**get template**コマンドを使用してください（詳細は[get core commands](#)を参照してください）。
2. 必要に応じて出力されたテンプレートを操作するために、**コマンドラインのJSONプロセッサ**、たとえば**jq**を使用してください。
3. **エンコード**コマンドを使用して、操作されたJSONをエンコードします（詳細を参照）。
4. エンコードされたJSONからオブジェクトを作成するには、**create**コマンドを使用してください。

例えば：

Bash

```
bw get template folder | jq '.name="My First Folder"' | bw encode | bw create folder
```

または

Bash

```
bw get template item | jq ".name=\"My Login Item\" | .login=$(bw get template item.login | jq '.use  
rname="jdoe" | .password="myp@ssword123"')" | bw encode | bw create item
```

成功した作成後、新しく作成されたオブジェクトはJSONとして返されます。

他のアイテムタイプを作成する

作成コマンドはデフォルトでログインアイテムを作成しますが、jqのようなコマンドラインJSONプロセッサを使用して、**.type=**属性を変更し、他のアイテムタイプを作成することができます。

お名前	値
ログイン	タイプ=1
セキュアメモ	タイプ=2
カード	タイプ=3
ID	タイプ=4

例えば、次のコマンドはセキュアメモを作成します：

Bash

```
bw get template item | jq '.type = 2 | .secureNote.type = 0 | .notes = "Contents of my Secure Not  
e." | .name = "My Secure Note"' | bw encode | bw create item
```

Note

上記の例で注意してください、セキュアメモはサブテンプレート (`.secureNote.type`) が必要です。アイテムタイプのサブテンプレートは、`bw get template`を使用して表示できます (詳細は[こちら](#)を参照してください)。

添付ファイルを作成する

添付ファイルを作成コマンドは、ファイルを既存のアイテムに添付します。

他の作成操作とは異なり、添付ファイルを作成するためにJSONプロセッサやエンコードを使用する必要はありません。代わりに、添付するファイルを指定するための `--file` オプションと、それを添付するアイテムを指定するための `--itemid` オプションを使用してください。例えば：

Bash

```
bw create attachment --file ./path/to/file --itemid 16b15b89-65b3-4639-ad2a-95052a6d8f66
```

Tip

あなたが使用したい `itemid` が正確にわからない場合は、`bw get アイテム` を使用してアイテムを返します (詳細を参照)、その `id` を含む。

取得する

`get` コマンドは、ポートから単一のオブジェクト (`item`、`username`、`password` など) を取得します。

Bash

```
bw get (item|username|password|uri|totp|exposed|attachment|folder|collection|organization|org-collection|template|fingerprint) <id> [options]
```

`get` コマンドは、項目 ID または文字列を引数として受け取ります。文字列を使用する場合 (例えば、正確な `id` 以外のもの)、`get` は一致する値を持つものをあなたの保管庫オブジェクトから検索します。例えば、次のコマンドは Github のパスワードを返します：

Bash

```
bw get password Github
```

Note

`get` コマンドは結果を一つだけ返すことができます、したがって、具体的な検索用語を使用するべきです。複数の結果が見つかった場合、CLI はエラーを返します。

添付ファイルを取得してください

添付ファイルを取得コマンドは、添付ファイルをダウンロードします。

Bash

```
bw get attachment <filename> --itemid <id>
```

添付ファイルを取得コマンドは、ファイル名と**正確なID**を取ります。デフォルトでは、**添付ファイルを取得**は現在の作業ディレクトリに添付ファイルをダウンロードします。異なる出力ディレクトリを指定するためには、**--output** オプションを使用できます。例えば：

Bash

```
bw get attachment photo.png --itemid 99ee88d2-6046-4ea7-92c2-acac464b1412 --output /Users/myaccount/Pictures/
```

Note

--outputを使用するとき、パスはディレクトリまたはファイル名を指定するために、スラッシュ(/)で終わらなければなりません(/Users/myaccount/Pictures/photo.png)。

メモを取得する

「**メモを取得**」コマンドは、任意の保管庫アイテムのメモを取得します。

Bash

```
bw get notes <id>
```

「**メモを取得**」コマンドは、正確なアイテムの**ID**または文字列を取ります。文字列を使用する場合（例えば、正確な**id**以外のもの）、**メモを取得**は、一致する値を持つ保管庫オブジェクトを検索します。例えば、次のコマンドはGithubのメモを返します：

Bash

```
bw get notes Github
```

テンプレートを取得する

「**get template**」コマンドは、オブジェクト（**アイテム**、**アイテム.field**、**アイテム.login**など）の予想されるJSON形式を返します：

Bash

```
bw get template (item|item.field|item.login|item.login.uri|item.card|item.identity|item.securenote|folder|collection|item-collections|org-collection)
```

あなたは `get template` を使用してフォーマットを画面に出力することができますが、最も一般的な使用例は、出力を `bw create` 操作にパイプすることです。これは、コマンドラインのJSONプロセッサ（例えばjq）と `bw encode` を使用して、テンプレートから取得した値を操作するためです。例えば：

Bash

```
bw get template folder | jq '.name="My First Folder"' | bw encode | bw create folder
```

Note

任意の `アイテム.xxx` テンプレートは、`アイテム` テンプレートのサブオブジェクトとして使用するべきです。例えば：

Bash

```
bw get template item | jq ".name=\"My Login Item\" | .login=$(bw get template item.login | jq '.username=\"jdoe\" | .password=\"myp@ssword123\"')\" | bw encode | bw create item
```

編集

`編集` コマンドは、保管庫内のオブジェクト（`アイテム`、`アイテム-コレクション` など）を編集します。

Bash

```
bw edit (item|item-collections|folder|org-collection) <id> [encodedJson] [options]
```

`編集` コマンドは、**正確な id**（編集するオブジェクト）とエンコードされたJSON（行うべき編集）を取ります。典型的なワークフローは次のようになるかもしれません：

1. `get` コマンドを使用して（[詳細を参照](#)）、編集するオブジェクトを出力します。
2. 必要に応じて出力されたオブジェクトを操作するために、コマンドラインのJSONプロセッサ、例えばjqを使用してください。
3. `エンコード` コマンドを使用して、操作されたJSONをエンコードします（[詳細を参照](#)）。
4. オブジェクトを編集するには、`編集` コマンド（オブジェクトの `id` を含む）を使用してください。

例えば、ログインアイテムのパスワードを編集するには：

Bash

```
bw get item 7ac9cae8-5067-4faf-b6ab-acfd00e2c328 | jq '.login.password="newp@ssw0rd"' | bw encode | bw edit item 7ac9cae8-5067-4faf-b6ab-acfd00e2c328
```

または、アイテムが含まれているコレクションを編集するには：

Bash

```
echo '["5c926f4f-de9c-449b-8d5f-aec1011c48f6"]' | bw encode | bw edit item-collections 28399a57-73a0-45a3-80f8-aec1011c48f6 --organizationid 4016326f-98b6-42ff-b9fc-ac63014988f5
```

または、コレクションを編集するには：

Bash

```
bw get org-collection ee9f9dc2-ec29-4b7f-9afb-aac8010631a1 --organizationid 4016326f-98b6-42ff-b9fc-ac63014988f5 | jq '.name="My Collection"' | bw encode | bw edit org-collection ee9f9dc2-ec29-4b7f-9afb-aac8010631a1 --organizationid 4016326f-98b6-42ff-b9fc-ac63014988f5
```

編集コマンドは、オブジェクトに置換操作を実行します。完了後、更新されたオブジェクトはJSONとして返されます。

リスト

「リスト」コマンドは、あなたの保管庫からオブジェクトの配列（アイテム、フォルダー、コレクションなど）を取得します。

Bash

```
bw list (items|folders|collections|organizations|org-collections|org-members) [options]
```

リストコマンドのオプションは、フィルタを使用して返される内容を指定します。これには、`--url`、`--folderid`、`--collectionid`、`--organizationid`、そして`--trash`が含まれます。どんなフィルターでも`null`または`notnull`を受け入れます。一つのコマンドで複数のフィルターを組み合わせると、OR操作が実行されます。例えば：

Bash

```
bw list items --folderid null --collectionid null
```

このコマンドは、フォルダーやコレクションにないアイテムを返します。

さらに、検索を使用して特定のオブジェクトを--検索 できます。フィルターと検索を一つのコマンドに組み合わせると、AND操作が実行されます。例えば：

Bash

```
bw list items --search github --folderid 9742101e-68b8-4a07-b5b1-9578b5f88e6f
```

このコマンドは、指定されたフォルダー内で文字列githubを含むアイテムを検索します。

削除

削除コマンドはあなたの保管庫からオブジェクトを削除します。削除は**正確なid**のみをその引数として取ります。

Bash

```
bw delete (item|attachment|folder|org-collection) <id> [options]
```

デフォルトでは、削除はアイテムをゴミ箱にSendし、そこに30日間保持されます。**-p**、**--permanent**オプションを使用して、アイテムを永久に削除することができます。

Bash

```
bw delete item 7063feab-4b10-472e-b64c-785e2b870b92 --permanent
```

org-collectionを削除するには、**--organizationid** も指定する必要があります。[組織ID](#)を参照してください。

⚠ Warning

削除を使用して削除されたアイテムは、最大30日間復元コマンドを使用して回復することができます（[詳細](#)を参照してください）。しかし、削除 **--永久**を使用して削除されたアイテムは**完全に削除され、回復不可能になります**。

復元する

復元コマンドは、ゴミ箱から削除されたオブジェクトを復元します。復元は、引数として**正確なID**のみを取ります。

Bash

```
bw restore (item) <id> [options]
```

例えば：

Bash

```
bw restore item 7063feab-4b10-472e-b64c-785e2b870b92
```

送る

Sendコマンドは、一時的な共有のためのBitwarden Sendオブジェクトを作成します。このセクションでは、シンプルなSend操作について詳しく説明しますが、sendは非常に柔軟性のあるツールであり、[CLIからのSendに関する専用の記事を参照することをお勧めします](#)。

簡単なテキストを送るために：

Bash

```
bw send -n "My First Send" -d 7 --hidden "The contents of my first text Send."
```

シンプルなファイルを作成してSend:

Bash

```
bw send -n "A Sensitive File" -d 14 -f /Users/my_account/Documents/sensitive_file.pdf
```

受け取る

受信コマンドは、Bitwarden Sendオブジェクトにアクセスします。Sendオブジェクトを受け取るには：

Bash

```
bw receive --password passwordforaccess https://vault.bitwarden.com/#/send/yawoill8rk6VM6zCATXv2A/9WN8wD-hzsDJjfnXLeNc2Q
```

組織のコマンド

組織ID

CLIから組織にアクセスするには、組織のID、個々のメンバーのID、およびコレクションのIDを知っている必要があります。

この情報をCLIから直接取得します。例： `bw list` を使用してください。

Bash

```
bw list organizations
bw list org-members --organizationid 4016326f-98b6-42ff-b9fc-ac63014988f5
bw list org-collections --organizationid 4016326f-98b6-42ff-b9fc-ac63014988f5
```

💡 Tip

あなたは `bw list` を使って、コレクションと `org-コレクション` の両方を表示できます。 `bw list collections` コマンドは、それらがどの組織に属しているかに関係なく、すべてのコレクションをリストします。 `bw list org-collections` は、 `--organizationid` を使用して指定された組織に属するコレクションのみをリストします。

移動する

📌 Note

2021年8月：共有コマンドは移動に変更されました。もっと詳しく知る

`move` コマンドは、Vault アイテムを組織に転送します。

Bash

```
bw move <itemid> <organizationid> [encodedJson]
```

「移動」コマンドは、コレクションIDをエンコードすることを要求し、正確な id (共有するオブジェクト) と正確な 組織id (オブジェクトを共有する組織) を取ります。例えば：

Bash

```
echo '["bq209461-4129-4b8d-b760-acd401474va2"]' | bw encode | bw move ed42f44c-f81f-48de-a123-ad01013132ca dfghbc921-04eb-43a7-84b1-ac74013bqb2e
```

完了後、更新されたアイテムが返されます。

確認する

`confirm` コマンドは、組織に招待されたメンバーが招待を受け入れたことを確認します。

Bash

```
bw confirm org-member <id> --organizationid <orgid>
```

「確認」コマンドは、正確なメンバーIDと正確な組織IDを取ります。例えば：

Bash

```
bw confirm org-member 7063feab-4b10-472e-b64c-785e2b870b92 --organizationid 310d5ffd-e9a2-4451-af87-ea054dce0f78
```

他のコマンド

設定

`config` コマンドは、Bitwarden CLIが使用する設定を指定します：

Bash

```
bw config server <setting> [value]
```

「bw config」の主な用途の一つは、自己ホスト型のBitwardenサーバーにCLIを接続することです。

Bash

```
bw config server https://your.bw.domain.com
```

💡 Tip

次のコマンドを実行して、BitwardenのEUサーバーに接続します:

Bash

```
bw config server https://vault.bitwarden.eu
```

値なしで**bw config server**を渡すと、接続しているサーバーを読み取ることができます。

ユニークな設定を持つユーザーは、各サービスのURLを個別に指定することを選択することができます。configコマンドの後続の使用はすべての前の仕様を上書きすることにメモしてください。したがって、変更を加えるたびにこれを一回のコマンドとして実行する必要があります。

Bash

```
bw config server --web-vault <url> \  
  --api <url> \  
  --identity <url> \  
  --icons <url> \  
  --notifications <url> \  
  --events <url> \  
  --key-connector <url>
```

📌 Note

bw config サーバー **--key-connector**組織でKey Connector を使用しており、マスター パスワードを削除した後に**--apikey**オプションを使用してログインしている場合は、このコマンドが必要です。

必要なURLを取得するために組織の所有者に連絡してください。

同期

同期コマンドは、Bitwardenサーバーから暗号化された保管庫をダウンロードします。このコマンドは、CLIでログインしてから他のクライアントアプリケーション（例えばウェブ保管庫、ブラウザの拡張機能、モバイルアプリ）でBitwardenの保管庫を変更したときに最も役立ちます。

Bash

```
bw sync
```

--last オプションを渡すことで、最後に同期が行われた時間のタイムスタンプ (ISO 8601) のみを返すことができます。

💡 Tip

それは重要です、同期はサーバーからのプルのみを実行することを覚えておくこと。あなたが保管庫に変更を加えるたび（例えば、作成、編集、削除）、データは自動的にサーバーにプッシュされます。

エンコード

エンコードコマンドはBase 64でstdinをエンコードします。このコマンドは通常、jqのようなコマンドラインJSONプロセッサと組み合わせて使用され、作成および編集操作を行う際に使用されます。例えば：

Bash

```
bw get template folder | jq '.name="My First Folder"' | bw encode | bw create folder

bw get item 7ac9cae8-5067-4faf-b6ab-acfd00e2c328 | jq '.login.password="newp@ssw0rd"' | bw encode |
bw edit item 7ac9cae8-5067-4faf-b6ab-acfd00e2c328
```

インポート

インポートコマンドは、Bitwardenのエクスポートまたは他のサポートされているパスワード管理アプリケーションからデータをインポートします。コマンドはファイルを指す必要があり、次の引数を含める必要があります：

Bash

```
bw import <format> <path>
```

例えば：

Bash

```
bw import lastpascsv /Users/myaccount/Documents/mydata.csv
```

💡 Tip

Bitwardenはインポートのための多くの形式をサポートしています、ここにリストするには多すぎます！あなたのCLIでリストを返すには、`bw インポート --formats`を使用してください、または[ここ](#)を見てください。

あなたがパスワードで作成した暗号化された.jsonファイルをインポートしている場合、インポートが完了する前にパスワードを入力するように求められます。

エクスポート

`エクスポート` コマンドは、保管庫のデータを `.json` または `.csv`、または暗号化された.jsonファイルとしてエクスポートします。

Bash

```
bw export [--output <filePath>] [--format <format>] [--password <password>] [--organizationid <orgid>]
```

デフォルトでは、`エクスポート` コマンドは現在の作業ディレクトリに `.csv` (`--format csv` を指定した場合と同等) を生成しますが、指定することもできます：

- `--format json` は、`.json` ファイルをエクスポートするためのものです。
- `--format encrypted_json` を使用して暗号化された.jsonファイルをエクスポートする
 - `--password` これは、あなたのアカウントの暗号化キーの代わりに、`encrypted_json` のエクスポートを暗号化するために使用するパスワードを指定します。
- `--output` 特定の場所にエクスポートするために
- `--raw` は、エクスポートをファイルではなく標準出力に返します。

組織の保管庫からのエクスポート

`エクスポート` コマンドを `--organizationid` オプションと共に使用することで、組織の保管庫をエクスポートできます。

Bash

```
bw export --organizationid 7063feab-4b10-472e-b64c-785e2b870b92 --format json --output /Users/myaccount/Downloads/
```

生成

`生成` コマンドは強力なパスワードまたはパスフレーズを生成します。

Bash

```
bw generate [--lowercase --uppercase --number --special --length <length> --passphrase --separator <separator> --words <words>]
```

デフォルトでは、**生成** コマンドは大文字、小文字、および数値を含む14文字のパスワードを生成します。これはパスすることと同等です。

Bash

```
bw generate -u!n --length 14
```

コマンドに利用可能なオプションを使用して、より複雑なパスワードを生成することができます。これには以下が含まれます：

- **--大文字, -u** (大文字を含む)
- **--小文字, -l** (小文字を含む)
- **--数値, -n** (数値を含む)
- **--特別な, -s** (特殊文字を含む)
- **--長さ** (パスワードの長さ、最小は5)

パスフレーズを生成します

生成 コマンドを**--パスフレーズ** オプションとともに使用すると、パスワードの代わりにパスフレーズを生成することができます。

Bash

```
bw generate --passphrase --words <words> --separator <separator>
```

デフォルトでは、**bw 生成 --passphrase** はダッシュ(-)で区切られた3語のパスフレーズを生成します。これはパスすることと同等です。

Bash

```
bw generate --passphrase --words 3 --separator -
```

コマンドに利用可能なオプションを使用して、複雑なパスフレーズを生成することができます。これには以下が含まれます：

- **--語** (語の数値)
- **--セパレータ** (セパレータ文字)
- **--capitalize, -c** (パスフレーズをタイトルケースにするために含める)

- `--数値を含める` (パスフレーズに数値を含める)

更新

`update` コマンドは、Bitwarden CLI が最新バージョンを実行しているかどうかを確認します。 `update` では、CLI が自動的に更新されません。

Bash

```
bw update
```

新しいバージョンが検出された場合、実行可能ファイルのための印刷された URL を使用して、または CLI をダウンロードするために使用したパッケージマネージャーで利用可能なツールを使用して、CLI の新しいバージョンをダウンロードする必要があります (例えば、`npm install -g @bitwarden/cli`) 。

ステータス

`status` コマンドは、設定されたサーバー URL、最後の同期のタイムスタンプ (ISO 8601)、ユーザーの電子メールと ID、ボルトのステータスなど、Bitwarden CLI に関するステータス情報を返します。

Bash

```
bw status
```

ステータスは、例えば、JSON オブジェクトとして情報を返します。

Bash

```
{
  "serverUrl": "https://bitwarden.example.com",
  "lastSync": "2020-06-16T06:33:51.419Z",
  "userEmail": "user@example.com",
  "userId": "00000000-0000-0000-0000-000000000000",
  "status": "unlocked"
}
```

「ステータス」は次のいずれかになる可能性があります：

- 「`unlocked`」、ログインしていてボルトのロックが解除されていることを示します (`BW_SESSION` キー環境変数はアクティブなセッション キーとともに保存されます)。
- 「`locked`」、ログインしているがボルトがロックされていることを示します (アクティブなセッション キーとともに `BW_SESSION` キー環境変数が保存されていません)。
- 「`unauthenticated`」、ログインしていないことを示します

 Tip

"status": "未認証"のとき、lastSync、 userEmail、そしてuserIDは常にnullを返します。

提供する

serveコマンドは、ローカルのエクスプレスウェブサーバーを起動し、HTTPインターフェースからのRESTful API呼び出しの形でCLIからアクセス可能なすべてのアクションを実行するために使用できます。

Bash

```
bw serve --port <port> --hostname <hostname>
```

デフォルトでは、serveはウェブサーバーをポート8087で起動しますが、--port オプションを使用して別のポートを指定することができます。

デフォルトでは、serveはあなたのAPIウェブサーバーをlocalhostにバインドしますが、--hostname オプションで別のホスト名を指定することができます。APIリクエストは、バインドされたホスト名からのみ行うことができます。

デフォルトでは、serveはOriginヘッダーを持つすべてのリクエストをブロックします。この保護は--disable-origin-protection オプションを使用して回避することができますが、これは推奨されません。

 Warning

ホスト名のバインディングを行わないために、--hostname allを指定することができますが、これによりネットワーク上の任意のマシンがAPIリクエストを行うことが可能になります。

API仕様を表示して、serveで呼び出しを行うためのヘルプを参照してください。

付録

グローバルオプション

次のオプションは全世界で利用可能です：

オプション

説明

--きれいに

出力形式を設定します。JSONは2つのスペースでタブが付けられています。

--生

詳細なメッセージの代わりに生の出力を返します。

--応答

レスポンス出力のJSON形式バージョンを返します。

オプション	説明
<code>--静か</code>	stdoutに何も返さないでください。たとえば、資格情報の値をファイルやアプリケーションにパイプするときに、このオプションを使用するかもしれません。
<code>--対話なし</code>	対話的なユーザー入力を求めないでください。
<code>--セッション</code>	環境変数から読み取る代わりにセッションキーを渡してください。
<code>-v、--バージョン</code>	Bitwarden CLIのバージョン数値を出力します。
<code>-h、--ヘルプ</code>	コマンドのヘルプテキストを表示します。

ZSHシェルの補完

Bitwarden CLIはZSHシェルの補完をサポートしています。シェル補完を設定するには、以下の方法のいずれかを使用します:

1. **バニラZSH:** 以下の行をあなたの`.zshrc`ファイルに追加してください:

Bash

```
eval "$(bw completion --shell zsh); compdef _bw bw;"
```

2. **バニラ (ベンダー補完):** 次のコマンドを実行してください:

Bash

```
bw completion --shell zsh | sudo tee /usr/share/zsh/vendor-completions/_bw
```

3. **zinit:** 次のコマンドを実行します:

Bash

```
bw completion --shell zsh > ~/.local/share/zsh/completions/_bw
zinit creinstall ~/.local/share/zsh/completions
```

自己署名証明書を使用する

あなたの自己ホスト型Bitwardenサーバーが自己署名TLS証明書を流出済みの場合、Node.js環境変数NODE_EXTRA_CA_CERTSを指定してください。

🍏 バッシュ

Bash

```
export NODE_EXTRA_CA_CERTS="absolute/path/to/your/certificates.pem"
```

🍷 パワーシェル

Bash

```
$env:NODE_EXTRA_CA_CERTS="absolute/path/to/your/certificates.pem"
```

列挙型

次の表は、文書化されたシナリオで必要な値を列挙しています:

2段階ログイン方法

どの二段階ログイン方法を使用するかを指定するために、ログインする際に使用します:

お名前	値
認証システム	0
Eメール	1
YubiKey	3

📌 Note

FIDO2とDuoはCLIに対応していません。

アイテムタイプ

作成コマンドで使用して、保管庫アイテムタイプを指定します:

お名前	値
ログイン	1
セキュアメモ	2
カード	3
ID	4

ログインURIのマッチタイプ

作成と編集コマンドで使用し、ログインアイテムのURIマッチ検出の動作を指定します：

お名前	値
ドメイン	0
ホスト	1
始まります	2
完全一致	3
正規表現	4
なし	5

フィールドタイプ

作成と編集のコマンドで使用してカスタムフィールドを設定します：

お名前	値
テキスト	0
非表示	1
真偽値	2

組織ユーザータイプ

ユーザーのタイプを示します：

お名前	値
オーナー	0
管理者	1
ユーザー	2
マネージャー	3
カスタム	4

組織ユーザーのステータス

ユーザーの組織内でのステータスを示します：

お名前	値
招待済み	0

お名前	値
承諾済み	1
確認済み	2
取り消し済み	-1