



Salesforce における多要素認証 (MFA) の適用について (2026/06/03)

【概要】

2026年6月22日に Salesforce の仕様変更に伴い、2026年6月22日より順次 MFA (多要素認証) が必須化されます (<https://help.salesforce.com/s/articleView?id=005321561&type=1>)。MFA 必須化に伴い、ミラロボのブラウザ操作コマンドによる Salesforce へのログイン手順が変わります。新しい手順では、Salesforce の API からログイン URL を発行し、その URL 経由でブラウザログインを行います。そのため、Salesforce の API を利用するようにシナリオを修正する必要があります。

本ドキュメントでは、Salesforce の API を利用するための OAuth 設定手順と、ミラロボでの連携方法についてご案内いたします。

なお、Salesforce ログインの新しい手順を行うためには、「JWT エンコード」という新しい RPA コマンドが、ミラロボおよび各種 OEM のアップデートによって、2026年6月11日に追加が予定されています。本手順では、追加予定の「JWT エンコード」コマンドを利用します。

【手順】

新手順は、Salesforce の API 利用準備を含め、大きく分けて3つのセクションがあります。

1. 証明書および秘密鍵の作成
2. Salesforce の OAuth 設定によって API 利用を許可
3. ミラロボのシナリオ修正

1. 証明書および秘密鍵の作成

ミラロボから Salesforce へログインするには、証明書 (server.crt) と秘密鍵 (client.pfx) という2つのファイルが必要になります。これらは、Salesforce の API 利用時に、API 利用者の身分証明のために利用します。下記の手順の通り、PowerShell を利用して作成します。

※ PowerShell は Windows で標準インストールされているソフトウェアです。

1.1 PowerShell を起動します

「Win」+「R」で表示されるダイアログに「powershell」と入力して Enter を押下します。



1.2 証明書を作成します

Powershell に下記の文字列を入力して Enter を押下します。

下記は、自己署名証明書を作成し、ファイルとして書き出すコマンドです。

```
$cert = New-SelfSignedCertificate `
-Subject "CN=SalesforceJWT" `
-CertStoreLocation "Cert:¥CurrentUser¥My" `
-KeyExportPolicy Exportable `
-KeySpec Signature `
-KeyAlgorithm RSA `
-KeyLength 2048 `
-HashAlgorithm SHA256 `
-NotAfter (Get-Date).AddYears(100)
Export-Certificate `
-Cert $cert `
-FilePath ".¥server.cer"
certutil `
-encode ".¥server.cer" ".¥server.crt"
```

項目	内容
New-SelfSignedCertificate	自己署名証明書を作成するコマンド。証明書は Windows の証明書ストアに保存されるため、ファイルとして保存するために「Export-Certificate」と「certutil」を利用する。証明書データは\$cert という参照 ID に一時的に保持する。
-Subject "CN=SalesforceJWT"	New-SelfSignedCertificate のオプション。 証明書の所有者名を「SalesforceJWT」に設定する。
-CertStoreLocation "Cert:¥CurrentUser¥My"	New-SelfSignedCertificate のオプション。 作成した証明書を、Windows の現在のユーザーの「個人」証明書ストアに保存する。
-KeyExportPolicy Exportable	New-SelfSignedCertificate のオプション。 秘密鍵をエクスポートできるように許可する。
-KeySpec Signature	New-SelfSignedCertificate のオプション。 鍵の用途を「デジタル署名用」に指定する。JWT 認証に必要な設定である。
-KeyAlgorithm RSA	New-SelfSignedCertificate のオプション。 暗号化アルゴリズムに「RSA」を使用する
-KeyLength 2048	New-SelfSignedCertificate のオプション。 鍵の長さを 2048 ビット（一般的な長さ）に設定する。



-HashAlgorithm SHA256	New-SelfSignedCertificate のオプション。 ハッシュアルゴリズムに「SHA-256」を指定する。
-NotAfter(Get-Date).AddYears(100)	New-SelfSignedCertificate のオプション。 証明書の有効期限を現在の日付から「100 年後」に設定する。有効期限は必要に応じて、設定してください。
Export-Certificate	証明書（公開鍵）をファイルとして書き出すコマンド。
-Cert \$cert	Export-Certificate のオプション。 変数 \$cert に格納した証明書を指定する
-FilePath ".\server.cer"	Export-Certificate のオプション。 現在操作中のフォルダに server.cer という名前で保存する。
certutil	ファイル形式を Salesforce が対応する crt 形式に変換するコマンド
-encode ".\server.cer" ".\server.crt"	certutil のオプション。 証明書ファイル (server.cer) から server.crt を作成

1.3 秘密鍵を作成します

秘密鍵パスワードは必要に応じて変更してください。

秘密鍵 (client.pfx) はログインに利用するファイルです。漏洩しないようご注意ください。

※"ここに秘密鍵のパスワードを記載してください" の部分は、ご自身で設定する任意のパスワードに書き換えてから実行してください。（例："TestPfx8495!" など）

<pre>\$pwd = ConvertTo-SecureString ` -String "ここに秘密鍵のパスワードを記載してください" ` -Force ` -AsPlainText Export-PfxCertificate ` -Cert \$cert ` -FilePath ".\client.pfx" ` -Password \$pwd</pre>	
項目	内容
ConvertTo-SecureString	指定のパスワードを安全な文字列（暗号文字列）に変換するコマンド。暗号文字列は\$pwd という参照 ID に一時的に保持する。
-String "ここに秘密鍵のパスワードを記載してください"	ConvertTo-SecureString のオプション。 設定したいパスワードの平文を指定する。
-Force	ConvertTo-SecureString のオプション。
-AsPlainText	平文を暗号文字列に変換するときに指定する。



Export-PfxCertificate	秘密鍵を指定の証明書から作成するコマンド。 PFX（PKCS #12）形式のファイルとして出力する
-Cert \$cert	Export-PfxCertificate のオプション。 証明証データを指定する。
-FilePath ".\client.pfx"	Export-PfxCertificate のオプション。 秘密鍵を client.pfx というファイル名で保存する
-Password \$pwd	Export-PfxCertificate のオプション。 秘密鍵にパスワードを設定する

ここで作成した証明書（server.crt）と秘密鍵（client.pfx）は後述の手順で利用しますので、ファイルの場所をメモしてください。

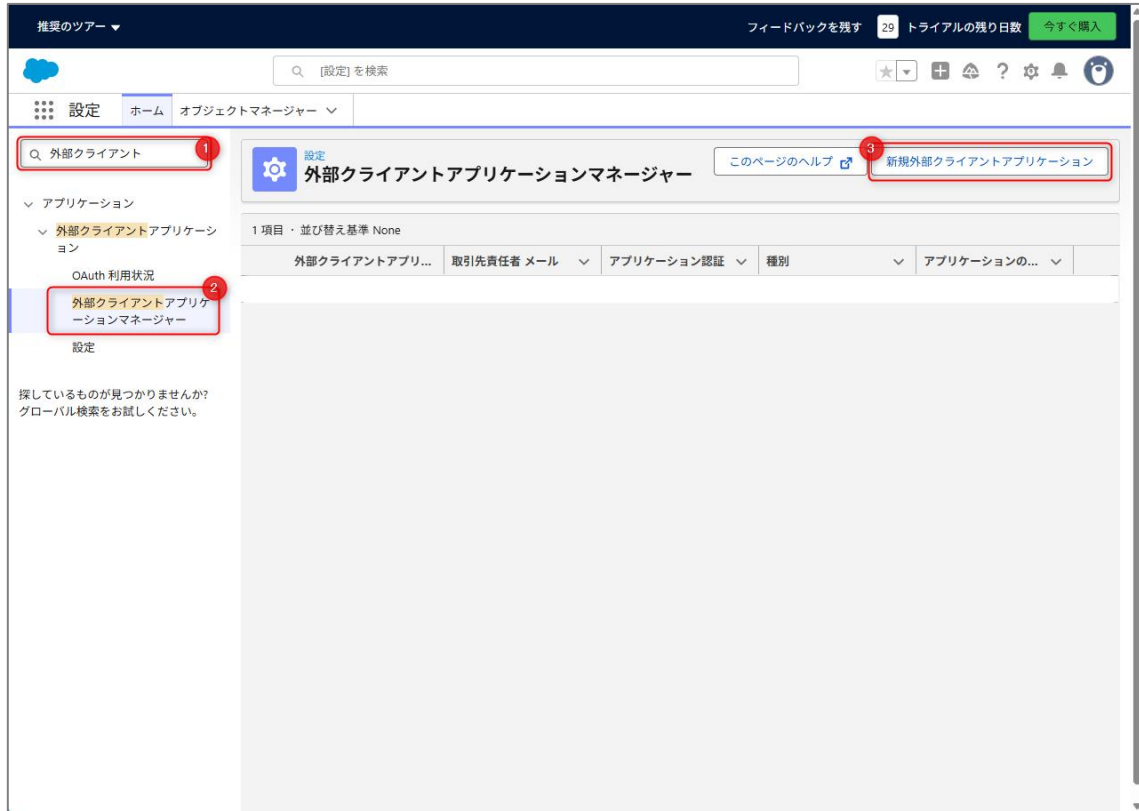
また、秘密鍵のパスワードはシナリオ内で設定する必要があるので、メモしてください。

2. Salesforce の OAuth 設定によって API 利用を許可

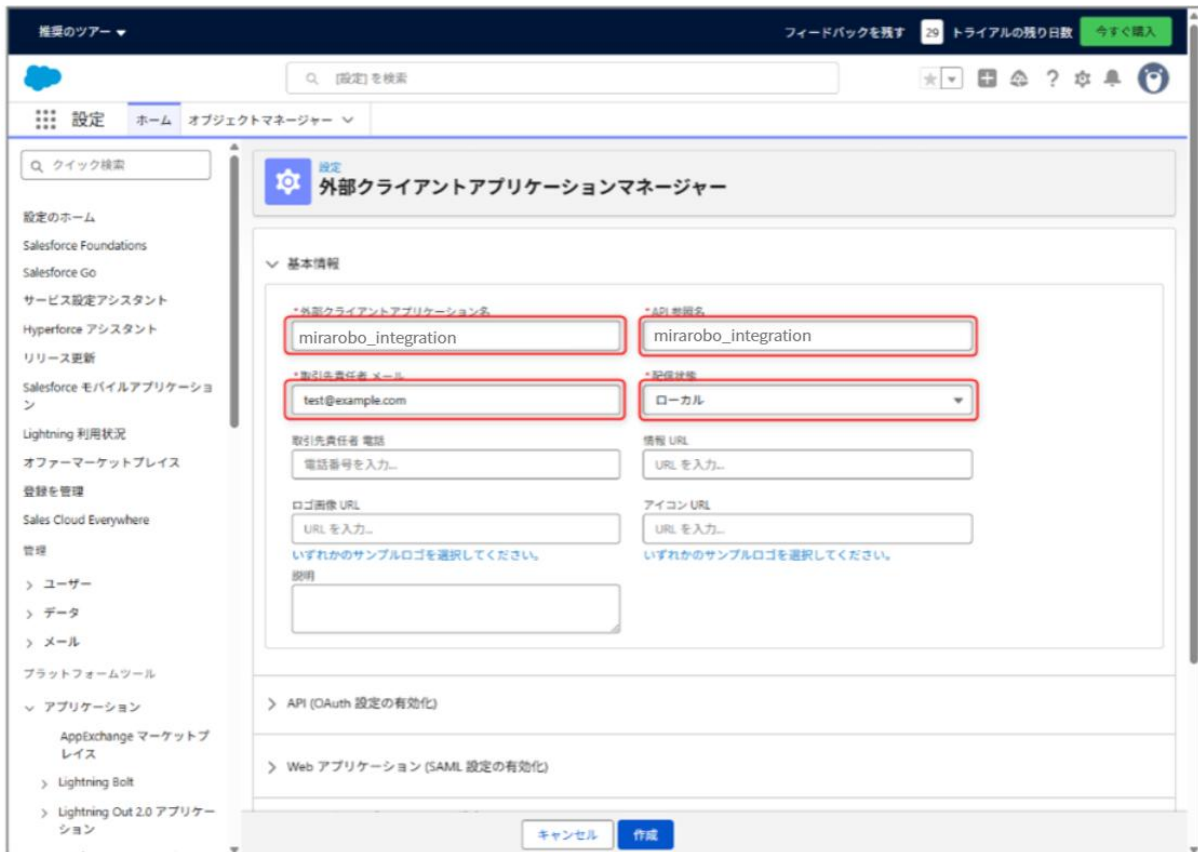
ミラロボとの連携に必要な外部クライアントアプリケーションの作成を行います。

2.1 Salesforce の画面にて外部クライアントアプリケーションの作成画面に移動します

The screenshot shows the Salesforce Lightning Setup One Home interface. The top navigation bar includes a search bar and a settings gear icon. A dropdown menu is open from the settings icon, with '設定' (Settings) highlighted. The main content area features several dashboards: '営業ホーム' (Business Home) with a total deal value of ¥72.9万, '私の取引先を計画' (Plan My Accounts) showing 3 accounts, '関係を深める' (Deepen Relationships) showing 2 accounts, 'パイプラインを作成' (Create Pipeline) showing 6 leads, '私の目標' (My Goals) for setting personal and monthly goals, and '今日の行動' (Today's Actions) with a forecast for the day.



2.2 基本情報を入力します





項目	内容
外部クライアントアプリケーション名	外部クライアントアプリケーションマネージャーに表示する外部クライアントアプリケーション名
API 参照名	プログラムからアプリケーションを参照するときの名前
取引先責任者メール	Salesforce がアプリケーション提供者またはそのサポートチームへの連絡時に使用する連絡先メール
配布状態	ローカル組織の外部クライアントアプリケーションの場合は「ローカル」に設定。 配布用の外部クライアントアプリケーションの場合は「パッケージ化済み」に設定。 今回は、ローカル組織の外部クライアントアプリケーションであるため「ローカル」に設定

2.3 API (OAuth 設定) を有効化します

OAuth 設定の有効化

OAuth を有効化

アプリケーション設定

* コールバック URL
https://localhost

* OAuth 範囲
利用可能な OAuth 範囲

Web ブラウザーを使用してユーザーデータを管理 (web)
いつでも要求を実行 (refresh_token, offline_access)

ID URL サービスにアクセス (id, profile, email, address, phone)
 API を使用してユーザーデータを管理 (api)
 フルアクセス (full)
 Connect REST API リソースにアクセス (chatter_api)
 Visualforce アプリケーションにアクセス (visualforce)
 一意のユーザー識別子にアクセス (openid)

すべてのトークンを調査
 ID トークンを設定

• Web ブラウザーを使用してユーザーデータを管理 (web)
• いつでも要求を実行 (refresh_token, offline_access)

フローの有効化
があるので選択して▶をクリック

クライアントログイン情報フローを有効化
 認証コードおよびログイン情報フローを有効化
 デバイスフローを有効化
 JWT ベアラーフローを有効化

証明書アップロード
証明書を選択

📎 ファイルをアップロード またはファイルをドロップ

トークン交換フローを有効化

キャンセル 作成



項目	内容
コールバック URL	認証が成功した後にユーザーのブラウザがリダイレクトされる URL。ミラロボでは <code>https://localhost</code> を設定
OAuth 範囲	ミラロボから Salesforce へログインできるようにするには、下記を追加。 <ul style="list-style-type: none">● Web ブラウザーを使用してユーザーデータを管理 (web)● いつでも要求を実行 (refresh_token, offline_access)
フローの有効化	ミラロボが対応する JWT ベアラーフローにチェック

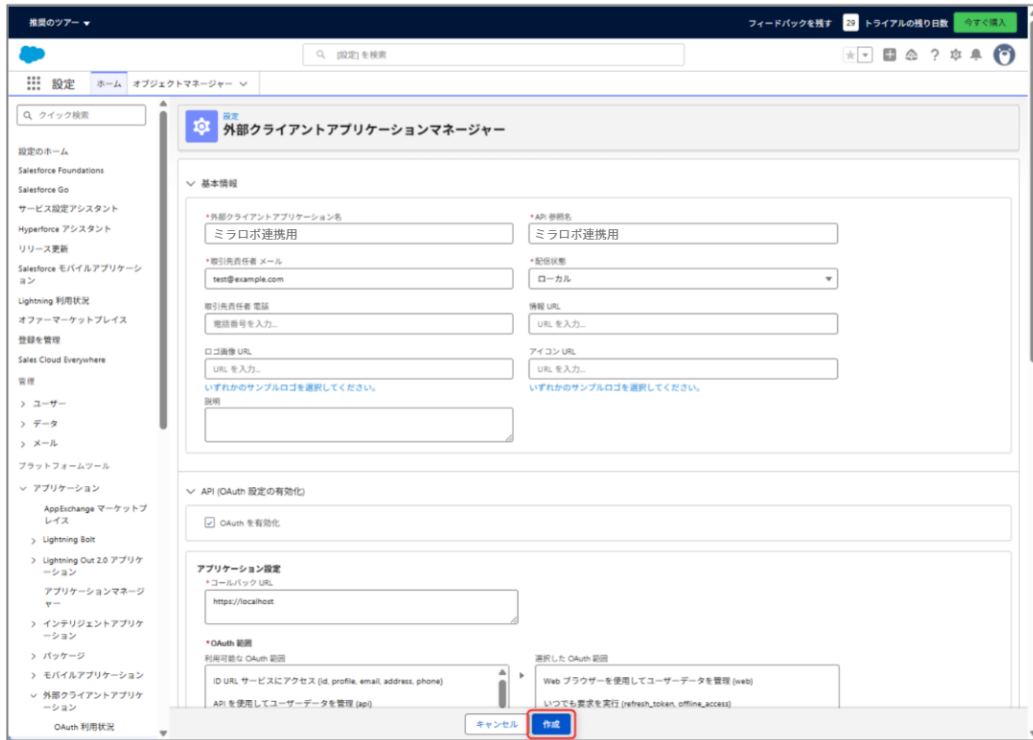
2.4 手順 1 で作成した証明書 (server.crt) をアップロードします

The screenshot shows the Salesforce OAuth settings interface. The 'API (OAuth settings activation)' section is expanded. Under 'Application Settings', the 'Callback URL' is set to 'https://localhost'. Under 'OAuth Scopes', 'Web browser to manage user data (web)' and 'Execute requests anytime (refresh_token, offline_access)' are selected. In the 'Flow Activation' section, the 'JWT bearer flow activation' checkbox is checked, and the 'Upload Certificate' button is highlighted with a red box.

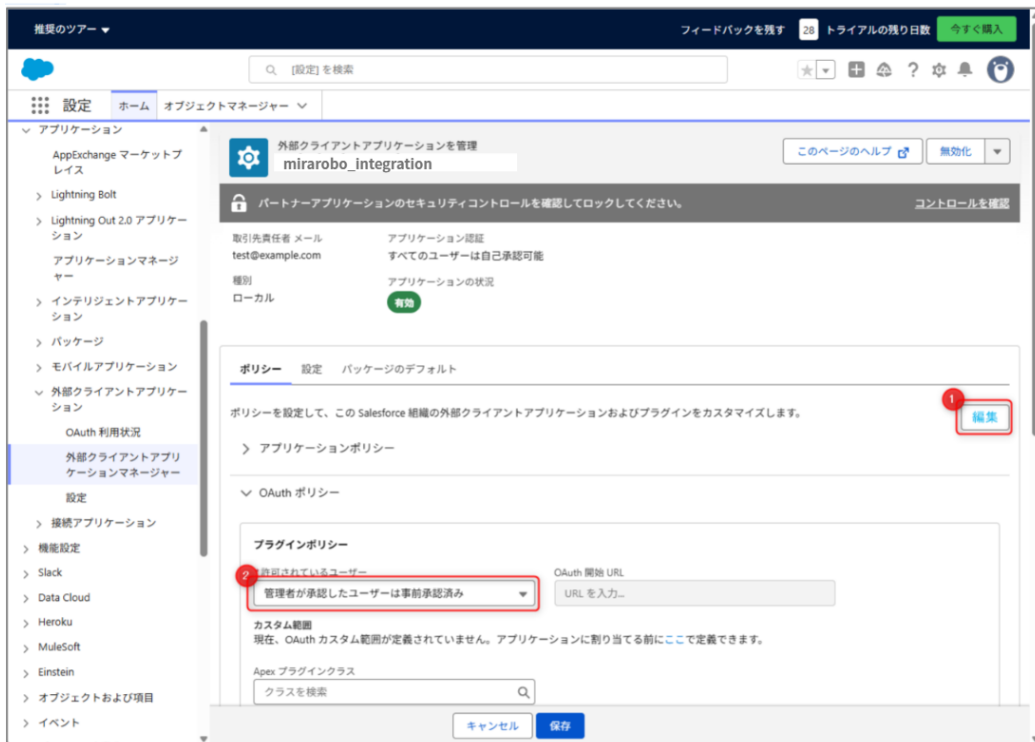
JWT ベアラーフローを有効化」にチェックを入れると、証明書アップロード欄が表示されます。手順 1.2 で作成した `server.crt` ファイルをアップロードしてください。



外部クライアントアプリケーションの設定が完了しましたので、「作成」をクリックします。



2.5 OAuth ポリシーを設定します



作成したアプリケーションの詳細画面で「ポリシー」タブを開き、「編集」をクリックします。



2.6 許可ユーザーの設定

項目	内容
許可されているユーザー	ミラロボからのログインを許可するユーザを設定する。 「すべてのユーザーは自己承認可能」と「管理者が承認したユーザーは事前承認済み」が選択可能だが、「すべてのユーザーは自己承認可能」は手動承認が必要であるため、「 管理者が承認したユーザーは事前承認済み 」を選択

2.7 アプリケーションポリシーを変更します

ミラロボからのログインを許可したユーザーに割り当てられているプロファイルを設定してください。

※プロファイルの確認方法は2枚目の画像に記載しています。

推奨のツアー ▼ フィードバックを残す 28 トライアルの残り日数 今すぐ購入

設定 ホーム オブジェクトマネージャー ▼

アプリケーション

- AppExchange マーケットプレイス
- Lightning Bolt
- Lightning Out 2.0 アプリケーション
- アプリケーションマネージャー
- インテリジェントアプリケーション
- パッケージ
- モバイルアプリケーション
- 外部クライアントアプリケーション
- OAuth 利用状況
- 外部クライアントアプリケーションマネージャー
- 設定
- 接続アプリケーション
- 機能設定
- Slack
- Data Cloud
- Heroku
- MuleSoft
- Einstein
- オブジェクトおよび項目
- イベント

ローカル 有効

ポリシー 設定 パッケージのデフォルト

ポリシーを設定して、この Salesforce 組織の外部クライアントアプリケーションおよびプラグインをカスタマイズします。 保存

アプリケーションポリシー

*開始ページ ①

なし

1 プロファイルを選択 該当するプロファイルを選択してください

選択可能なプロファイル

- Analytics Cloud Integration User
- Analytics Cloud Security User
- Chatter External User

選択済みプロファイル

- システム管理者
- 標準ユーザー

権限セットを選択

利用可能な権限セット

- Constraint Rules Engine Licenseless permission set
- Sales User
- sfdc.activityplatform

選択済みの権限セット

キャンセル 保存 3



3. ミラロボのシナリオ修正

<全体の流れ>

The screenshot displays a sequence of 10 steps in a scenario editor:

- データの記憶 (文字)**: 文字: ここにログインユーザ名 (ボタン: ユーザ名)
- パスワードの記憶 (秘密鍵のパスワードを入力してください)**: パスワード設定方法: 直接入力 (ボタン: 秘密鍵パスワード)
- 現在の時刻を記憶**: 時刻の形式: [UNIX時間], 先頭の0削除: 無し (ボタン: 時刻)
- 計算結果を記憶 (このシナリオでは5分間 (300秒)、URLを有効化します。)**: 演算式: 時刻 + 300 (ボタン: 有効期限)
- JSON Web Token生成 (コンシューマー鍵を設定してください)**: (ボタン: エンコードされた認証情報)
- Web API (Salesforce側に認証リクエストを送信します)**: メソッド: POST, URL: https://login.salesforce.com/services/oa ... (ボタン: [応答ステータス], [応答ヘッダ], [応答データ])
- JSON値取得**: 取得元データ参照ID: [応答データ], 取得値の型: オブジェクト (Object), キー名: access_token (ボタン: アクセストークン)
- JSON値取得**: 取得元データ参照ID: [応答データ], 取得値の型: オブジェクト (Object), キー名: instance_url (ボタン: インスタンスURL)
- ブラウザ起動 (ログイン済みの状態でSalesforceが開きます)**: ブラウザ: アシロボブラウザ, 開始URL: インスタンスURL /secur/frontdoor.jsp?sid= アクセストークン, ウィンドウ最大化: 有り (ボタン: ブラウザ)
- 通知**: Salesforceのサイトが表示されるまでに時間がかかる場合があります。必要に応じて待機を入れてください。

コマンド	内容
1. データを記憶	ログインユーザ名を設定します。
2. パスワードの記憶	秘密鍵のパスワードを設定します。
3. 現在の時刻を記憶	UNIX 時間という形式で現在時刻を取得します。 ※ログイン URL の発行には有効期限を設定する必要があり、その形式が UNIX 時間である必要があるため、UNIX 時間形式で取得します。
4. 計算結果を記憶	URL を有効化する時間を設定します。 ※UNIX 時間は数値です。有効化したい秒数を加算すると、その秒数有効化できます。例えば、5 分間 (300 秒) 有効にしたい場合は、+300 します。
5. JSON Web Token 生成	身分証明データを作成します。
6. Web API	身分証明データを利用して、Salesforce へログインします。



7. JSON 値取得	ログインに成功すると、アクセストークンと URL が返ってきますので、アクセストークンを取得します。
8. JSON 値取得	ログインに成功すると、アクセストークンと URL が返ってきますので、URL を取得します。
9. ブラウザ起動	アクセストークンと URL を利用して、ログイン済みの状態で Salesforce を開きます。

6 番目の WebAPI コマンドで Salesforce の API を利用します。API で JWT データを送信することで、ログイン用の URL を発行できます。ログイン済みの状態で Salesforce を開いた後は、これまで通りブラウザ操作コマンドで操作できます。現在のログインを行っている箇所を置き換えていただきますようお願い致します。

API 経由でログイン URL が発行されますので、その URL にブラウザ起動コマンド等でアクセスする形になります。URL には有効期限がありますので、必要に応じて 4 番目の計算結果を記憶コマンドを利用して有効期限を延長してください。

詳細コマンド解説

■データを記憶

1 データの記憶 (文字) ✕
文字: *****@*****.com ユーザー名

salesforce ログインユーザー名を記憶します。(設定>ユーザー からご確認いただけます)

■パスワードの記憶

2 パスワードの記憶 ✕
パスワード設定方法: 直接入力 秘密鍵

ご自身で設定した、秘密鍵のパスワードを記憶します。

■現在の時刻を記憶

3 現在の時刻を記憶 ✕
時刻の形式: [UNIX時間], 先頭の0削除: 無し 時刻

UNIX 時間とは、協定世界時 (UTC) の 1970 年 1 月 1 日午前 0 時 0 分 0 秒からの経過秒数で表す、コンピュータ上での時刻表現のことです。ログイン URL の発行には有効期限設定が必要です。その形式が UNIX 時間である必要があるため、UNIX 時間形式で取得します。



■計算結果を記憶

4 計算結果を記憶 (5分間) ✕
演算式: 時刻 + 300 有効期限

URL を有効化する時間を設定します。

※UNIX 時間は数値です。有効化したい秒数を加算すると、その秒数有効化できます。

例えば、5 分間（300 秒）有効にしたい場合は、+300 します。

■JSON Web Token 生成

事前に登録した証明書とコンシューマー鍵を利用し、Salesforce へ安全に認証要求を行うための身分証明データを作成します ※秘密鍵ファイル→client.pfx を指定する

5 JSON Web Token生成 (salesforceへ安全に認証要求を行うための身分証明データを作成) ✕
エンコードされた認証情報

Salesforce へのログインには主に「Web API」コマンドと「JWT エンコード」コマンドを利用します。

「JWT エンコード」コマンドは、Salesforce の API を利用する際の身分証明データを作成するために使います。身分証明データを標準形式（JSON Web Token）に変換するコマンドです。

JWT エンコードコマンドと WebAPI コマンドの設定画面、サンプルシナリオは下記の通りです。

Json Web Tokenに変換

データ+

Key	iss	☰	Value	ここにコンシューマー鍵	☰	✕
Key	sub	☰	Value	\${ユーザ名}	☰	✕
Key	aud	☰	Value	https://login.salesforce.com	☰	✕
Key	exp	☰	Value	\${有効期限}	☰	✕

秘密鍵C://Users/...☰

パスワード設定方法パスワード参照IDから▼

パスワード参照IDPFXパスワード▼

データ参照IDエンコードされた認証情報

[詳細表示]

メモコンシューマー鍵と秘密鍵を設定してください

✕ キャンセル

✓ OK



項目	内容
データ	JWT 形式に変換するデータを指定する。 ※Salesforce が指定する項目については下記に記載しています。
秘密鍵	データ変換に利用する秘密鍵を指定する。 拡張子は下記に対応している。 .pem .key .pfx .p12
パスワード設定方法	パスワード設定の方法を指定する。 以下の設定方法から選択できる。 1. パスワード無し 2. 直接入力 3. パスワード参照 ID 手順 1.3 で指定したパスワードを設定する
データ参照 ID	JWT 変換したデータを保持するデータ参照 ID 名を定義する

Salesforce が指定する身分証明データの項目

項目	内容
iss	証明書を登録した外部クライアントアプリケーション（もしくは接続アプリケーション）の コンシューマー鍵
sub	ログインするユーザ名
aud	認証サーバの URL。利用している環境に合わせて設定 ※本番環境、Develop Edition https://login.salesforce.com ※Sandbox 環境 https://test.salesforce.com ※Experience Cloud https://site.force.com/customers
exp	認証情報の有効期限。時間の形式は UNIX 時間



■ Web API

身分証明データを salesforce へ送信し、認証結果としてアクセストークン(Salesforce にログイン済みであることを証明する一時的な通行証)を取得します。

7 Web API (身分証明データを利用して、Salesforceへログイン)

メソッド: POST URL: https://login.salesforce.com/services/oa ...

[応答ステータス] [応答ヘッダ] [応答データ]

Web API

メソッド	POST		
テンプレート	自由入力		
URL	https://login.salesforce.com/services/oauth2/token		
クエリパラメータ	+		
ヘッダ	+		
ボディ	<input checked="" type="radio"/> Form-Data <input type="radio"/> JSON <input type="radio"/> Text		
Form-Data	+		
Key	grant_type	Value	urn:ietf:params:oauth:grant-type:jwt-bearer
Key	assertion	Value	\${エンコードされた認証情報}
タイムアウト (秒)	10		
メモ	Salesforce側に認証リクエストを送信します		

【注意】API仕様については各サービスにお問い合わせください。

× キャンセル

✓ OK

項目	内容
メソッド	リクエストメソッドを指定します。 Salesforce へは POST で送信する必要がありますので、POST を設定します。
URL	リクエストの送信先を設定します。環境に合わせて設定してください。 ※本番環境、Develop Edition



	<p>https://login.salesforce.com/services/oauth2/token</p> <p>※Sandbox 環境 https://test.salesforce.com/services/oauth2/token</p> <p>※Experience Cloud https://site.force.com/customers/services/oauth2/token</p>
ボディ	ここでは Form-Data を選択します。Text でも送信可能です。
Form-Data	<p>Salesforce に送信するデータを指定します。</p> <p>grant_type: urn:iETF:params:oauth:grant-type:jwt-bearer assertion: エンコードされた認証情報</p> <p>※grant_type の値は Salesforce の仕様により、値が決まっています。</p>

■JSON 値取得

ログインに成功すると、アクセストークンと URL が返ってきますので、それぞれのコマンドで、アクセストークン、URL を取得します。

```
7  JSON値取得
取得元データ参照ID: [応答データ], 取得値の型: オブジェクト (Object) , キー名: アクセストークン
access_token

8  JSON値取得
取得元データ参照ID: [応答データ], 取得値の型: オブジェクト (Object) , キー名: インスタンスURL
instance_url
```

■ブラウザ起動

取得したアクセストークンを利用して、ログイン済みの状態で Salesforce を開きます。

```
9  ブラウザ起動 (ログイン済みの状態でsalesforceが開きます)
ブラウザ: ミラロボブラウザ
開始URL: インスタンスURL /secur/frontdoor.jsp?sid= アクセストークン
ウィンドウ最大化: 有り
ブラウザ
```