インフォマート API 利用における OAuth2.0 認証手順

作成日 2017 年 6 月 14 日
更新日 2020 年 7 月 21 日

株式会社インフォマート

更新内容

更新日時	対象項目	内容
2017/06/14	-	新規作成
2020/07/21	⑤期限切れアクセストークンの再発行	レスポンスのボディ形式を指定可能に変更

はじめに

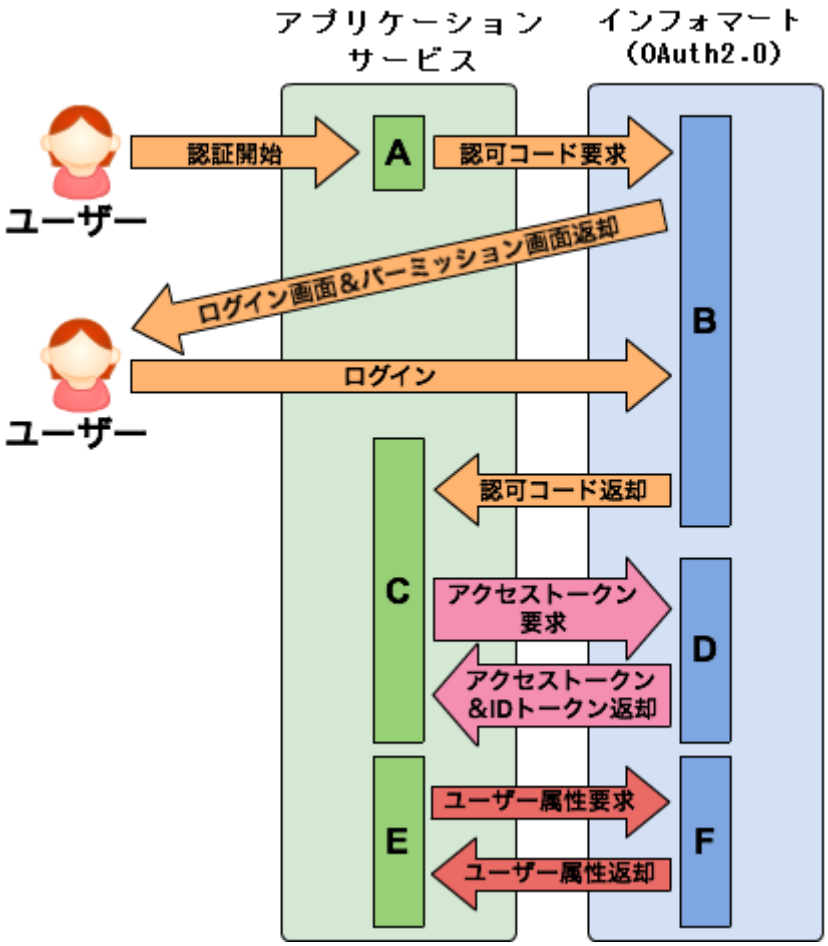
インフォマート API の呼び出しには、OAuth2.0 による認証を受ける必要があります。
OAuth2.0 を使うことで、インフォマート API を利用するサービスは、インフォマートプラットフォーム ID（※1 以下、PFID）とパスワードを保存したり処理したりすることなく、PFID を使用した BtoB プラットフォームサービスを利用することが出来ます。
本文書では、インフォマート API における OAuth2.0 認証の手続き、及び利用手順を説明します。

※1 インフォマートプラットフォーム ID とは
インフォマートの BtoB プラットフォーム（受発注や請求書）にログインする際に入力するメールアドレス形式の ID のことです。

1.OAuth2.0 の概要

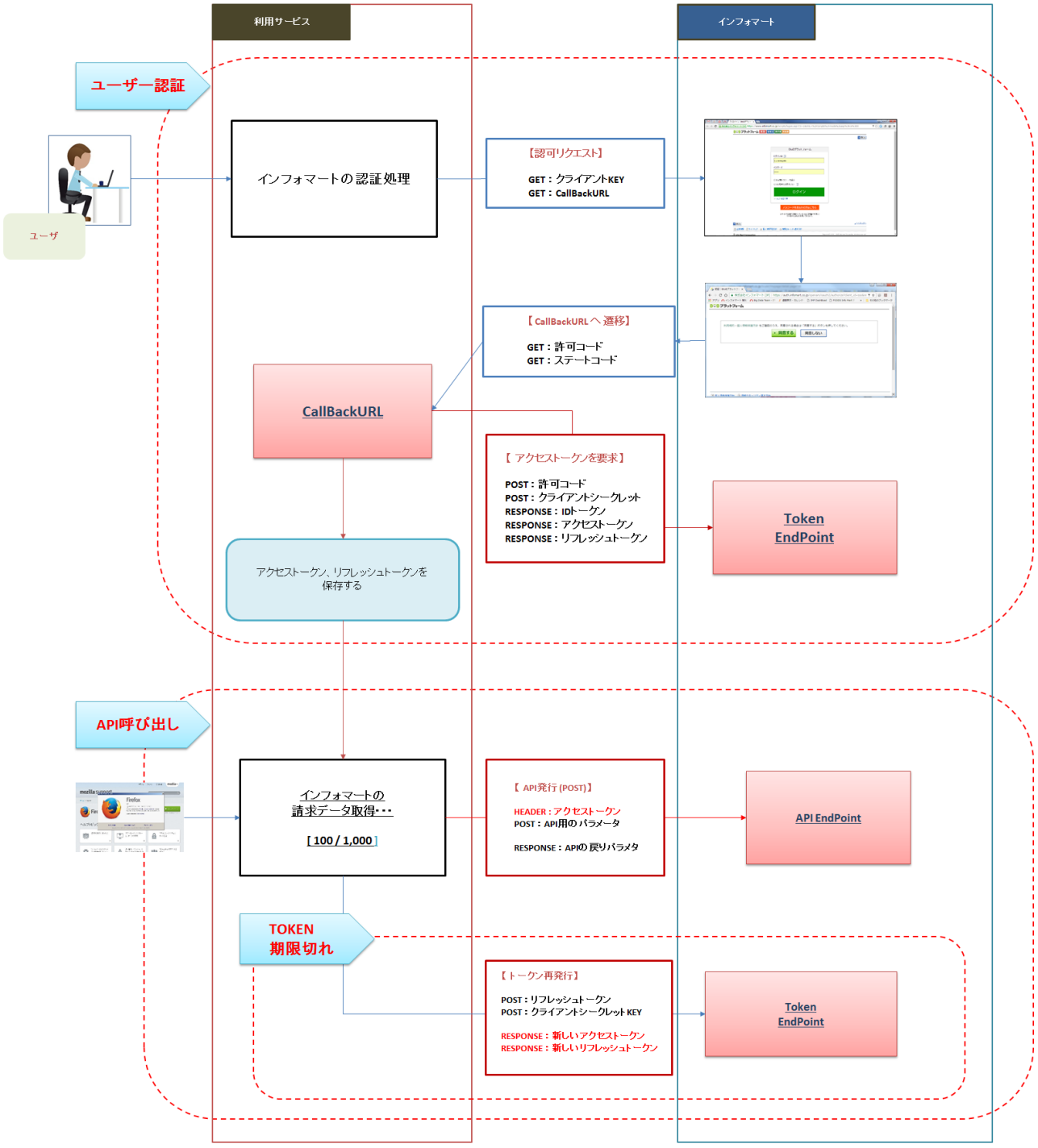
OAuth とは、サードパーティーアプリケーションによる HTTP サービスへのアクセスを可能にする認可フレームワークです。

サードパーティーアプリケーションでは、インフォマートの提供する OAuth2.0 を利用することで、ユーザの ID やパスワードをサービスの中に保持する必要なくインフォマートの認可情報を利用することができます。



OAuth2.0 概念図

2.インフォマートにおける OAuth2.0 認証フロー



① 認可リクエスト

インフォマート API を利用するユーザのブラウザに、PFID の認証を行うためのページを表示させるためのリクエスト URL になります。
ユーザはこの認証ページにて、PFID のユーザ ID とパスワードを入力すると、利用サービスが用意しているコールバック URL のページへリダイレクトします。

認可リクエストの URL

・本番環境

https://auth.infomart.co.jp/openam/oauth2/authorize?realm=/api

・テスト環境

http://authtest.infomart.co.jp/openam/oauth2/authorize?realm=/api

認可リクエスト入力パラメータ

パラメータ	名前	説明
client_id	クライアント ID	必須。サービスの申請時に発行されたクライアント ID。
redirect_uri	コールバック URL	必須。PFID の認証後、リダイレクトされるページの URL。 サービスの申請時に指定した URL と一致する必要がある。
response_type	レスポンスタイプ	必須。許可コードをリクエストするために“code”を指定する。
state	ステートコード	任意の文字列。ユーザの認可後に行われるリダイレクト時にこの値が含まれます。
scope	スコープ	必須。認証情報のアクセスについて要求する範囲を指定する。インフォマート API を利用する場合は、下記を固定で指定。 “openid profile email qualified”
access_type	アクセス種別	リフレッシュトークンを取得するために、下記を設定する。 “offline”

リクエストサンプル

```
https://auth.infomart.co.jp/openam/oauth2/authorize?realm=/api&
client_id=sample_client_id&
redirect_uri=http://xxxx/callback&
response_type=code&
scope=openid%20profile%20email%20qualified&
state=sample_state_code&
access_type=offline
```

② コールバック URL

コールバック URL とは、インフォマートが ID 認証を行ったユーザのブラウザに対して、許可コードとステートコードとともにリダイレクトする URL です。

許可コードは、インフォマート API を利用するためのアクセストークンを発行するために使用します。許可コードの有効期間は 120 秒のため、有効期限が切れる前にアクセストークンを発行する必要があります。アクセストークンの発行については後述します。

ステートコードは、認可リクエストを発行した際に設定した値がそのまま返却されます。認可リクエストの値と一致することを確認することで、他者が成りすましで認可リクエストを発行したりすることを防ぎ、セキュリティが向上します。

コールバック URL への出力パラメータ

パラメータ	名前	説明
code	許可コード	ID 認証が行われた際に発行されるコード。 アクセストークンの取得に使用する。
state	ステートコード	認可リクエストに設定した値がそのまま設定される。

リダイレクト URL のイメージ

```
http://xxxx/callback?
state=sample_state_code&
code=e1a82b2b-c612-4c2c-8a15-a2afc8e7b743
```

③ アクセストークンの発行

認可リクエスト後、コールバック URL に渡される認可コードとサービス申請時に発行された、インフォマート API を利用するためのアクセストークンとリフレッシュトークンを発行します。

アクセストークンとリフレッシュトークンは利用サービス内で保存して再利用することで、継続的にインフォマート API を利用することができます。

アクセストークンの有効期間は 5 分となっており、アクセストークンの有効期限が切れた場合、リフレッシュトークンを使用してアクセストークンの再発行をする必要があります。アクセストークンの再発行は後述します。

アクセストークンの発行は Token EndPoint に対して、application/x-www-form-urlencoded 形式のリクエストを送信します。

Token EndPoint の URL

・本番環境

https://auth.infomart.co.jp/openam/oauth2/access_token?realm=/api

・テスト環境

http://authtest.infomart.co.jp/openam/oauth2/access_token?realm=/api

アクセストークン・リクエストパラメータ

パラメータ	名前	説明
grant_type	権限種別	”authorization_code”を設定。
code	許可コード	コールバック URL で取得した許可コード。
redirect_uri	コールバック URL	許可リクエストで使用したコールバック URL
client_id	クライアント ID	サービス申請時に発行されたクライアント ID
client_secret	クライアントシークレット ID	サービス申請時に発行されたクライアントシークレット ID

リクエストイメージ

```
POST https://auth.infomart.co.jp/openam/oauth2/access_token?realm=/api HTTP/1.1
Host: auth.infomart.co.jp
Content-Type: application/x-www-form-urlencoded

grant_type=authorization_code&
code=e1a82b2b-c612-4c2c-8a15-a2afc8e7b743&
redirect_uri=https%3A%2F%2Fxxxxx%2Ecallback&
client_id=sample_client_id&
client_secret=sample_secret
```

アクセストークン・レスポンス内容

項目	説明
scope	アクセストークンの持つスコープ
expires_in	アクセストークンの有効期限（秒）
token_type	アクセストークンの種類。”Bearer”固定。
access_token	発行したアクセストークン。
refresh_token	アクセストークンの再発行に使用するリフレッシュトークン。

アクセストークン・レスポンスサンプル

```
{"scope":"openid profile email qualified",
"expires_in":300,
"token_type":"Bearer",
"access_token":"db36aeb3-021e-4a51-aaa6-b94c0da4542e",
"refresh_token":"f21b49b0-31b0-4441-8cc9-0ef7e14ae738"}
```

④ インフォマート API 呼び出し

インフォマート API を呼び出す場合は、発行したアクセストークンを認証リクエストヘッダーに設定することで、認証を行った PFID でインフォマート API の処理を実行することができます。
リクエストヘッダーには下記の形式で設定してください。

Authorization: Bearer ACCESS_TOKEN

インフォマート API の URL、パラメータ等は各サービスのインフォマート API の仕様書をご確認ください。

インフォマート API リクエストサンプル

```
POST https://api.infomart.co.jp/sample_api?prm=x HTTP/1.1
Host: api.infomart.co.jp
Authorization: Bearer db36aeb3-021e-4a51-aaa6-b94c0da4542e
Content-Type: text/xml;charset=UTF-8
```

⑤ 期限切れアクセストークンの再発行

アクセストークンの有効期限は 5 分となっており、有効期限の切れたアクセストークンは認証に使用できません。アクセストークンの有効期限が切れた場合は、アクセストークンの発行時に同時に発行されているリフレッシュトークンを用いて、アクセストークンの再発行をする必要があります。

アクセストークンの再発行を行うと、リフレッシュトークンも同時に再発行されます。リフレッシュトークンの有効期限は 31 日となっており、有効期限が切れる前にリフレッシュトークンを再発行し保持しておく必要があります。

アクセストークンの再発行は Token EndPoint に対して、application/x-www-form-urlencoded 形式のリクエストを送信します。

Token EndPoint の URI

・本番環境

https://auth.infomart.co.jp/openam/oauth2/access_token?realm=/api

・テスト環境

http://authtest.infomart.co.jp/openam/oauth2/access_token?realm=/api

アクセストークン・リクエストパラメータ（再発行）

パラメータ	名前	説明
grant_type	権限種別	”refresh_token”を設定。
refresh_token	リフレッシュトークン	アクセストークンに対応するリフレッシュトークン
client_id	クライアント ID	サービス申請時に発行されたクライアント ID
client_secret	クライアントシークレット ID	サービス申請時に発行されたクライアントシークレット ID
response_type	レスポンスのボディ形式	“json”または”xml”を指定(指定されなかった場合は”json”として扱う)。※認可リクエストの response_type とは意味が異なるので注意。

リクエストサンプル

```
POST https://auth.infomart.co.jp/openam/oauth2/access_token?realm=/api HTTP/1.1
Host: auth.infomart.co.jp
Content-Type: application/x-www-form-urlencoded

grant_type=refresh_token&
refresh_token= f21b49b0-31b0-4441-8cc9-0ef7e14ae738&
client_id=sample_client_id&
client_secret=sample_secret&
response_type=json
```

アクセストークン・レスポンス内容

項目	説明
scope	アクセストークンの持つスコープ
expires_in	アクセストークンの有効期限（秒）
token_type	アクセストークンの種類。”Bearer”固定。
access_token	発行したアクセストークン。
refresh_token	アクセストークンの再発行に使用するリフレッシュトークン。

アクセストークン・レスポンスサンプル(リクエストで response_type=json を指定したとき)

{"scope":"openid profile email qualified",
"expires_in":3600,
"token_type":"Bearer",
"access_token":"db36aeb3-021e-4a51-aaa6-b94c0da4542e",
"refresh_token":"f21b49b0-31b0-4441-8cc9-0ef7e14ae738"}

アクセストークン・レスポンスサンプル(リクエストで response_type=xml を指定したとき)


```
<?xml version='1.0' encoding='UTF-8'?>
<root_element>
    <access_token>db36aeb3-021e-4a51-aaa6-b94c0da4542e</access_token>
    <refresh_token>f21b49b0-31b0-4441-8cc9-0ef7e14ae738</refresh_token>
    <scope>openid profile email qualified</scope>
    <token_type>Bearer</token_type>
    <expires_in>3600</expires_in>
</root_element>
```

3.認証処理 実装サンプル（Java）

OAuth2.0 の仕様どおりであれば実装のプログラム言語、使用するモジュールは問いません。
今回は Java プログラムにおける実装サンプルを提示します。

サンプルでは、Spark Framework と Google URL Shortener API を使用しています。

```
public class Sample {

    public static void main(String... args) {

        setPort(8080);

    /**
     * OAuth 認証を行う
     * クライアント ID とコールバック URL を指定して、OAuth 認証ページへリダイレクトする
     */
    get("/auth", (Request request, Response response) -> {

        // 認証 EndPoint とクライアント ID を指定
        AuthorizationCodeRequestUrl codeUrl
                = new AuthorizationCodeRequestUrl(
                        "https://auth.infomart.co.jp/openam/oauth2/authorize?realm=/api",
                        "infomart_test");
        // リクエストするスコープをセット
        codeUrl.setScopes(Arrays.asList("openid profile qualified"));
        // 認証 EndPoint から許可コードを要求
        codeUrl.setResponseTypes(Arrays.asList("code"));
        // 任意でセッションごとに別々となるステートコードをセット（セキュリティ担保のため）
        codeUrl.setState("this_is_test_state_code");
        // nonce をセット（
        codeUrl.set("nonce", "this_is_one_time_phrase");
        // RefreshToken を併せて要求
        codeUrl.set("access_type", "offline");
        // コールバック URL を設定
        codeUrl.setRedirectUri("http://localhost:8080/api/callback.page");

        // 認証 EndPoint へリダイレクト
        response.redirect(codeUrl.build());
        return null;

    });
```

```
        /**
         * コールバック処理
         * ① 許可コードから AccessToken と RefreshToken を要求する
         * ② この場合は、そのまま AccessToken を利用できるが、
         *    AccessToken は有効期限を短く切っているため、
         *    サンプルとして、あえて RefreshToken を利用して最新の AccessToken を再発行させる
         * ③ AccessToken を組み込んで API を呼ぶ
         */
        get("/api/callback.page", (Request request, Response response) -> {

            /**
             * ①の処理
             */

            // 許可コードとステートコードを取得
            String allowCode = request.queryParams("code");
            String stateCode = request.queryParams("state");

            // 許可コードが自身の設定したものかを確認することで
            // セキュリティの強度が上がるが、ここでは処理は割愛する

            /**
             * 許可コードを利用して AccessToken と RefreshToken を要求
             */
            AuthorizationCodeTokenRequest tokenUrl = new AuthorizationCodeTokenRequest(
                    new NetHttpTransport(),
                    new JacksonFactory(),
                    new GenericUrl("https://auth.infomart.co.jp/openam/oauth2/access_token?realm=/api"),
                    allowCode
            );
            // authorization_code を指定
            tokenUrl.setGrantType("authorization_code");
            // コールバック URL を設定
            tokenUrl.setRedirectUri("http://localhost:8080/api/callback.page");
            // クライアント ID を設定
            tokenUrl.set("client_id", "infomart_test");
            // クライアントシークレット ID を設定
            tokenUrl.set("client_secret", "XXXXXXXXXX");


            TokenResponse tr = null;
            String accessToken = null;
            String refreshToken = null;
            String idToken = null;
            try {
                tr = tokenUrl.execute();
                accessToken = tr.getAccessToken();
                refreshToken = tr.getRefreshToken() == null ? "null" : tr.getRefreshToken();
                idToken = (String)tr.get("id_token");

            } catch (IOException e) {
            } finally {
                tr = null;
            }
```

（次頁に続く）

```
            /**
             * ②の処理
             */
            TokenRequest treq = new TokenRequest(
                new NetHttpTransport(),
                new JacksonFactory(),
                new GenericUrl("https://auth.infomart.co.jp/openam/oauth2/access_token?realm=/api"),
                "refresh_token"
            );
            treq.set("client_id", "infomart_test");
            treq.set("client_secret", "XXXXXXXXX");
            treq.set("refresh_token", refreshToken);
            TokenResponse tres = null;
            try {
                tres = treq.execute();
                // 再発行された AccessToken
                accessToken = tres.getAccessToken();
                // ReffeshToken も再発行のセットであることに注意すること
                refreshToken = tres.getRefreshToken();
            } catch (IOException e) {
            } finally {
                tres = null;
            }

            /**
             * ③の処理
             */
            StringBuilder response_all = new StringBuilder();
            try {

                HttpURLConnection con = null;
                // インフォマート API へのリクエスト
                URL url = new URL("https://api.infomart.co.jp/sampleapi");
                con = (HttpURLConnection)url.openConnection();
                con.setRequestMethod("GET");

                // リクエストヘッダーへのトークン埋め込み
                con.setRequestProperty("Authorization", "Bearer " + accessToken);
                con.connect();

                try {
                    try (BufferedReader reader =
                                new BufferedReader(new InputStreamReader(con.getInputStream(), "UTF-8"))) {
                        String res;
                        while ((res = reader.readLine()) != null) response_all.append(res);
                    } catch (IOException ex) {
                    }
                } finally {
                    con.disconnect();
                }

            } catch (MalformedURLException ex) {
                Logger.getLogger(Sample.class.getName()).log(Level.SEVERE, null, ex);
            } catch (IOException ex) {
                Logger.getLogger(Sample.class.getName()).log(Level.SEVERE, null, ex);
            }

            return "AccessToken == " + accessToken + "<br>"
                    + "RefreshToken == " + refreshToken + "<br><br>"
                    + response_all.toString();
        });

    }
}
```