Node.jsからGoogle Calendarの予定を取得する

公開日: 2024 / 01 / 07

目次

やりたいこと😎

Node.js でコードを実行して、Google API を通じてカレンダーから予定を取得したい。最終的には定期的にコードを実行して毎週月曜日に 1 週間分予定とかを取得したいけど、一旦それは置いておく。

認証情報の取得😈

公式ドキュメント

https://developers.google.com/calendar/api/guides/overview?hl=ja

ドキュメントに従って認証の情報を作成🧑‍🎨

Node.js のクイックスタート

に従いながら、**「デスクトップアプリケーションの認証情報を承認する」**までを完了する。このクイックスタートはブラウザを使って認証を完了する方法なので、サーバーからリクエストを送りたいというツールとは違ってきます。ダウンロードした `credentials.json` の中身の情報を使っていきます。

リフレッシュトークンの取得☕

サーバーからリクエストを送る際の認証にアクセストークンを用います。

ただ、アクセストークンは有効な期間が定まっており、おそらく Google API では 1 時間です。

これでは、アクセストークンの期限が切れるたびに、リフレッシュトークンを用いて手動で発行し直さなければなりません。

ただ、`googleapis` の npm パッケージでは、リフレッシュトークンを設定しておくと自動でアクセストークンの再発行を行ってくれるので、今回はこれを利用します。

以下の記事を参考にしました。

https://cloud.google.com/apigee/docs/api-platform/security/oauth/access-tokens?hl=ja

https://blog.shinonome.io/google-api/

認可コードの取得💯

https://cloud.google.com/apigee/docs/api-platform/security/oauth/access-tokens?hl=ja#requestingan authorizationcode

リフレッシュトークンを取得するにはまず認可コードを取得します。

必要なものは `credentials.json` の中身の

- クライアント ID

です。

ブラウザで

```
https://accounts.google.com/o/oauth2/auth?
client_id=取得したクライアントID
&redirect_uri=http://localhost <= なんでもいいのでlocalhost
&scope=https://www.googleapis.com/auth/calendar.readonly
&access_type=offline
&response_type=code
```

この URL にアクセスします。すると、OAuth 2.0 サーバーはユーザーを認証し、アプリケーションが要求するスコープにアクセスするために、ユーザーの同意を得る必要があります。その上でリダイレクトされた先の URL に含まれる「`code`」というクエリパラメーターが認可コードの値です。

アクセストークン（リフレッシュトークン）の取得🥹

```
curl \
-d code=さっき取得した認可コード \
-d client_id=最初に取得したクライアントID \
-d client_secret=最初に取得したクライアントシークレット \
-d redirect_uri=http://localhost \
-d grant_type=authorization_code \
https://accounts.google.com/o/oauth2/token
```

curl コマンドを使ってリクエストを送ります。すると

```
{
  "access_token": アクセストークン,
  "expires_in": 3599,
  "refresh_token": リフレッシュトークン,
  "scope": "https://www.googleapis.com/auth/calendar.readonly",
  "token_type": "Bearer"
}
```

このようなレスポンスが返ってきます。これでリフレッシュトークン取得完了です。

次は今取得した

- クライアント ID
- クライアントシークレット
- リフレッシュトークン

を用いて Node.js でコードを実行してみましょう！

サンプルコード🎥

ライブラリの追加

https://github.com/googleapis/google-api-nodejs-client

このライブラリをインストールしましょう。自分は 2024 年 1 月 7 日時点でバージョン `126.0.0` を使用しています。

サンプルコード

```
import dotenv from 'dotenv';
import { google } from 'googleapis';

dotenv.config();

// 環境変数から認証情報を取得
const clientId = process.env.GOOGLE_OAUTH_CLIENT_ID;
const clientSecret = process.env.GOOGLE_OAUTH_CLIENT_SECRET;
const refreshToken = process.env.GOOGLE_OAUTH_REFRESH_TOKEN;
const calendarId = process.env.CALENDAR_ID;

// 認証のための関数
const getGoogleOAuth = async () => {
  const googleOAuth = new google.auth.OAuth2(
    clientId,
    clientSecret,
    'http://localhost',
  );

  // 取得のリクエスト時に新しいアクセストークンを取得
  googleOAuth.setCredentials({
    refresh_token: refreshToken,
  });

  try {
    const accessTokenResponse = await googleOAuth.getAccessToken();

    const accessToken = accessTokenResponse.token;

    if (!accessToken)
      throw new Error('有効なアクセストークンを取得できませんでした');

    googleOAuth.setCredentials({
      access_token: accessToken,
    });

    return googleOAuth;
  } catch (err) {
    console.log('エラーの中身', err);
    throw new Error('アクセストークン取得時にエラーが発生しました');
  }
};

/**
 * @param timeMin new Date().toISOString() etc...
 * @param timeMax new Date().toISOString() etc...
 */
export const getEventListFromGoogleCalendar = async (
  timeMin: string,
  timeMax: string,
) => {
  try {
    // 認証
    const googleOAuth = await getGoogleOAuth();

    const calendar = google.calendar({ version: 'v3', auth: googleOAuth });

    const res = await calendar.events.list({
      calendarId,
      timeMin,
      timeMax,
      timeZone: 'Asia/Tokyo',
    });

    if (!res.data.items)
      throw new Error('正常にイベントを取得できませんでした');

    return res.data.items;
  } catch (err) {
    console.log('カレンダーからイベント取得時にエラーが発生しました', err);
  }
};
```

カレンダー ID🗓️

参照するカレンダーの ID も必要になります。

https://qiita.com/mikeneko_t98/items/80e26434149208d44fe5

こんな感じで調べることができます。

リフレッシュトークンを oauth2Client にセットする🔑

```
googleOAuth.setCredentials({
  refresh_token: refreshToken,
});
```

この部分でリフレッシュトークンを設定することで、アクセストークンの再生成を自動でやってくれるようになります。

実行結果🥳

```
[
  {
    kind: 'calendar#event',
    etag: '"hoge"',
    id: 'hoge',
    status: 'confirmed',
    htmlLink: 'hoge',
    created: '2023-12-27T11:47:23.000Z',
    updated: '2023-12-27T11:48:21.781Z',
    summary: '予定の名前',
    colorId: '5',
    creator: { email: 'hoge@gmail.com', self: true },
    organizer: { email: 'hoge@gmail.com', self: true },
    start: { dateTime: '2024-01-13T11:00:00+01:00', timeZone: 'Asia/Tokyo' },
    end: { dateTime: '2024-01-13T15:00:00+09:00', timeZone: 'Asia/Tokyo' },
    recurrence: [ 'RRULE:FREQ=MONTHLY;BYDAY=2SA' ],
    iCalUID: 'hoge@google.com',
    sequence: 1,
    reminders: { useDefault: true },
    eventType: 'default',
  },
];
```

こんな感じで予定が取れました！

最後に

結構むずかった…

では

Bye

TypeScript Node.js