

クロスドメインアクセス で利用

2024-08-23

記事中の「<https://hgotoh.jp>」を「<https://wiki.hgotoh.jp>」へ変更してください。以前のドメインから移動したので記事通りでは動作しなくなりました。

概要

クロスドメインアクセスで処理が動かないという利用者が結構いるようなのでサンプルを書いてみました。

このサンプルの前提としてAssistantSeikaとブラウザは同じPCで実行しているものとします。HTTP機能は有効化され、待ち受けアドレスをlocalhostにした状態です。

事前設定

発声させるため **Access-Control-Allow-Origin** ヘッダの内容に「<https://hgotoh.jp>」を指定します。

発声しない	発声する
 <p>AssistantSeika 20200820/u</p> <p>使用製品 基本設定 HTTP機能設定 話者一覧</p> <p>HTTP機能設定</p> <p><input checked="" type="checkbox"/> HTTP機能を利用する</p> <p>待ち受けアドレス localhost</p> <p>待ち受けポート 7180</p> <p>ワークフォルダ C:%Work</p> <p>ドキュメントルートフォルダ C:%Work%app</p> <p>CORSヘッダ</p> <p>Access-Control-Allow-Origin http://localhost:7180</p> <p>Access-Control-Allow-Headers Content-Type</p> <p>ユーザID SeikaServerUser</p> <p>パスワード SeikaServerPassword</p> <p>待ち受けURL http://localhost:7180</p> <p>バージョンURL http://localhost:7180/VERSION</p> <p>WebUI URL http://localhost:7180/webui/</p> <p>起動する</p> <p>※設定値を変更したら「起動する」ボタンを押してください</p>	 <p>AssistantSeika 20200820/u</p> <p>使用製品 基本設定 HTTP機能設定 話者一覧</p> <p>HTTP機能設定</p> <p><input checked="" type="checkbox"/> HTTP機能を利用する</p> <p>待ち受けアドレス localhost</p> <p>待ち受けポート 7180</p> <p>ワークフォルダ C:%Work</p> <p>ドキュメントルートフォルダ C:%Work%app</p> <p>CORSヘッダ</p> <p>Access-Control-Allow-Origin https://hgotoh.jp</p> <p>Access-Control-Allow-Headers Content-Type</p> <p>ユーザID SeikaServerUser</p> <p>パスワード SeikaServerPassword</p> <p>待ち受けURL http://localhost:7180</p> <p>バージョンURL http://localhost:7180/VERSION</p> <p>WebUI URL http://localhost:7180/webui/</p> <p>起動する</p> <p>※設定値を変更したら「起動する」ボタンを押してください</p>

サンプルUI

このページ「

<https://hgotoh.jp/wiki/doku.php/documents/voiceroid/assistantseika/interface/http/http-004>」からPCで実行しているAssistantSeikaへリクエストを送るサンプルです。

ボタンを押すと、テキストボックスに入力されたcidの話者で発声させるリクエストがAssistantSeikaに送られ発声します。
自分の環境にあるAssistantSeikaのユーザー、パスワードCIDを指定してみてください。

ユーザID:
パスワード:
cid: アカネちゃんカワイイヤッター

説明

ブラウザのセキュリティにより、サーバA から取得したコンテンツ内から サーバB へのアクセスは制限が課されます。この制限を突破できるとクロスサイトスクリプティングと言われる話になっていきます。

ですが、サーバBに「サーバAのコンテンツからの参照を許す」と許可をしてもらう事でブラウザのセキュリティは制限を解除します。このあたりの[記事](#)を参照してみてください。
これはブラウザのセキュリティのため、ブラウザじゃないプログラムでは有効ではありません。

つまり、AssistantSeikaのAccess-Control-Allow-Origin に “ <https://hgotoh.jp> ” を設定することで
AssistantSeika は “ <https://hgotoh.jp> ” のコンテンツから参照されることを許可します」と宣言するようになります。

サンプルコード

サンプルのHTMLコードは以下のようになっています。

```
<html>
<body>
ユーザID:<input id="uid" type="text" value="SeikaServerUser" /><br/>
パスワード:<input id="pass" type="text" value="SeikaServerPassword" /><br/>
cid:<input id="cid" type="text" value="2001" /> <button id="kawaii">アカネ
ちゃんカワイイヤッター
</button>

<script>
  async function calltts()
  {
    var bodyJson    = {"talktext": "アカネちゃんカワイイヤッター"};
    var headers     = {
      'Authorization' : 'Basic ' +
btoa(unescape(encodeURIComponent( document.getElementById('uid').value + ":"
+ document.getElementById('pass').value ))),
      'Content-Type'  : 'application/json'
    };
    var fetchopt    = { mode: 'cors', credentials: 'include', method:
'POST', body: JSON.stringify(bodyJson), headers: headers};
    var res         = await fetch("http://localhost:7180/PLAY2/" +
document.getElementById('cid').value , fetchopt);
```

```
    var json = await res.json();
  }

  document.getElementById('kawaii').addEventListener('click', calltts);
</script>
</body>
</html>
```

fetch APIで直接AssistantSeikaを叩くので、ブラウザによっては認証情報入力のタイミングがありません。なのでfetchのリクエストヘッダに認証ヘッダ(Authorizationヘッダ)を追加しています。
.....UID+Passじゃない方法を考えたほうが良さげかな。

[技術資料](#), [AssistantSeika](#), [HTML](#), [JavaScript](#), [CORS](#)

From:

<https://wiki.hgotoh.jp/> - 努力したWiki

Permanent link:

<https://wiki.hgotoh.jp/documents/tools/assistantseika/interface/http/http-004>

Last update: **2024/08/23 00:44**

