Windows 版 telnet を使って
GET 通信、POST 通信を体験する
Ver 1.0

Seiichi Nukayama

2022 年 9 月 10 日

目次

1 準備

1.1 サーバー側

以下のファイルをサーバー側ドキュメントルートに置く。

- get_receive.php
  GET 通信用
- index.php
  サーバーに接続したことがわかるようにする。
- post_receive.php
  POST 通信用
- sample.html
  ダウンロード用ファイル
- c.png
  画像ダウンロード用
- choki.png
  画像ダウンロード用
- favicon.ico
  なくてもかまわない。

以下、サーバーの IP アドレスを “192.168.31.27” とする。ポート番号を 8080 とする。

1.2 クライアント側

Windows 版 telnet.exe を “c:¥Windows¥system32” に用意する。telnet をダウンロードすれば、ここにインストールされる。

2 普通の GET リクエスト

コマンドプロンプトにて、以下のコマンドを入力する。

```
> telnet 192.168.31.27 8080
```

すると、コマンドプロンプト画面には何も表示されなくなる。これは、サーバーに接続しているのである。これ以降、ここで入力した文字はすべてサーバーに送られるため画面では見えなくなる。

したがって、TeraPad などのエディタで文字列を用意しておき、それをこの画面に貼り付けるとよい。

また、貼り付けるのも、Ctrl + v は効かない。Ctrl キーも、v キーも、すべてサーバーに送られるからである。コマンドプロンプトの左上のアイコンのメニューの” 編集” — “貼り付け” を使う。

エディタで以下の文字列を用意する。

```
GET /sample.html HTTP/1.1
Host: 192.168.31.27
```

これを貼り付ける。そして、"Enter" キーを 1 回、あるいは、もう 1 回押下する。"空行" が終わりの印である。

これは" ヘッダ部" で、GET 通信では ヘッダ部だけを送ることになっている。ここでは、ドキュメントルートにある sample.html ファイルを要求している。また、HTTP/1.1 プロトコルでの通信を求めている。

すると、以下のような文字列が送られてくるはずである。

```
HTTP/1.1 200 OK
Date: Sun, 14 Aug 2022 13:44:15 GMT
Server: Apache/2.4.54 (Debian)
Last-Modified: Sun, 14 Aug 2022 13:43:12 GMT
ETag: "210-5e633b24c11af"
Accept-Ranges: bytes
Content-Length: 528
Vary: Accept-Encoding
Content-Type: text/html

<!doctype html>
<html lang="ja">
  <head>
    <meta charset="utf-8"/>
    <meta name="viewport" content="width=device-width,initial-scale=1"/>
    <title>sample</title>
  </head>
  <body>
    <h1>サンプル</h1>
    <p>これは、サンプルのhtmlです。</p>
    <div id="btn">クリックしてね</div>
    <script>
    'use strict';

     document.getElementById('btn').onclick = function() {
       alert('クリックしたね')
     }
    </script>
  </body>
</html>
```

これは、動作しているサーバーによって多少の違いがあるかもしれない。途中の空行より上が" ヘッダ部" で、空行より下が" ボディ部" である。

ヘッダ部の先頭が" ステータス行" で、クライアントからの要求に対する返答である。ここでは、"HTTP/1.1" ぷろとこるでの通信要求と、"sample.html" のダウンロード要求に対して "OK" を返答している。

ヘッダ部の一番下に "Content-Type: text/html" とあるが、空行の後に送る文字列のタイプをクライアントのブラウザに伝えている。

空行の後が、ボディ部で、これがダウンロードされて、ブラウザにて描画 (レンダリング) される。

もし、HTML 文の中に <img src="choki.png">タグがあれば、その画像がブラウザによってサーバーに 以下のように GET リクエストされる。

```
1 GET /choki.png HTTP/1.1
2 Host: 192.168.1.17
```

これは、以下のようにレスポンスされる。

```
HTTP/1.1 200 OK
Date: Sat, 10 Sep 2022 04:02:59 GMT
Server: Apache/2.4.54 (Debian)
Last-Modified: Thu, 18 Aug 2022 21:43:44 GMT
ETag: "5f26-5e68ae025c400"
Accept-Ranges: bytes
Content-Length: 24358
Content-Type: image/png

  PNG

IHDR    $       pHYs


MiCCPPhotoshop ICC profilex  SwX    >    eVB    l

... (以下、略) ...
```

ヘッダ部の最下行は以下のようになっている。

`Content-Type: image/png`

画像として png 画像を送ることをブラウザに伝えている。空行のあとに送られているのが、画像データである。

3 GET リクエスト (クエリ文字列)

3.1 英字の場合

GET リクエストでクエリ文字列を送ることができる。

`telnet 192.168.31.27 8080` としたあとの画面で以下を入力する。

```
1 GET /get_receive.php?name=Taro HTTP/1.1
2 Host: 192.168.31.27
```

これは、サーバーのドキュメントルートにある get_receive.php をダウンロード要求している。そして、そのファイルに対して `name=Taro` という文字列を渡している。

サーバー側では、まず get_receive.php が PHP プログラムとして動作し、name という名前で “Taro” という文字列を受け取る。その後、PHP は HTML 文字列を生成し、それをクライアントに返す。

以下のようなレスポンスとなる。

```
1  HTTP/1.1 200 OK
2  Date: Sat, 10 Sep 2022 04:25:43 GMT
3  Server: Apache/2.4.54 (Debian)
4  X-Powered-By: PHP/7.4.30
5  Vary: Accept-Encoding
6  Content-Length: 354
7  Content-Type: text/html; charset=UTF-8
8
9  <!doctype html>
10 <html lang="ja">
11   <head>
12     <meta charset="utf-8"/>
```

```
    <meta name="viewport" content="width=device-width,initial-scale=1"/>
    <title>get_receive</title>
  </head>
  <body>
    <h1>get receive</h1>
    <p>name is Taro</p>
    <script>
    'use strict';

    </script>
  </body>
</html>
```

クライアントによって送られた “name=Taro” は、 name is Taro という HTML 文字列となって埋め込まれている。

3.2 日本語文字列を送る場合

GET リクエストにおいて、日本語文字列をクエリ文字列として送る場合は、“URL エンコード” しなくてはならない。

たとえば、以下のサイトで URL エンコード できる。

URL エンコード・デコード

“桃太郎” を URL エンコードすると、“%E6%A1%83%E5%A4%AA%E9%83%8E” となる。これを以下のように GET リクエストする。

```
GET /get_receive.php?name=%E6%A1%83%E5%A4%AA%E9%83%8E HTTP/1.1
Host: 192.168.31.27
```

以下のようなレスポンスが返ってくる。

```
HTTP/1.1 200 OK
Date: Sun, 14 Aug 2022 14:02:37 GMT
Server: Apache/2.4.54 (Debian)
X-Powered-By: PHP/7.4.30
Vary: Accept-Encoding
Content-Length: 359
Content-Type: text/html; charset=UTF-8

<!doctype html>
<html lang="ja">
  <head>
    <meta charset="utf-8"/>
    <meta name="viewport" content="width=device-width,initial-scale=1"/>
    <title>get_receive</title>
  </head>
  <body>
    <h1>get receive</h1>
    <p>name is 桃太郎taro</p>
    <script>
    'use strict';

    </script>
  </body>
```

```
</html>
```

実際は、ブラウザにて、日本語文字列をユーザーがいちいち URL エンコードしなくてもいいように、ブラウザが URL エンコードしてサーバーに GET リクエストしてくれている。

4 POST リクエスト

4.1 英字の場合

POST リクエストの場合、送信すべき文字列はボディ部の中に書く。そして、何バイトなのかをヘッダ部に書く。以下のようになる。

```
POST /post_receive.php HTTP/1.1
Host: 192.168.31.27
Content-Length:13
Content-Type: application/x-www-form-urlencoded

name=momotaro
```

今度は、ドキュメントルートの “post_receive.php” を指定している。“name=momotaro” という文字列を送信している。13 バイトである。

それと、フォームで送信するので、Content-Type: に application/x-www-form-urlencoded という記述が必要になる。

このようにフォームでは、ボディ部に送信文字列を記述しているのである。

これに対するレスポンスは以下である。

```
HTTP/1.1 200 OK
Date: Sat, 10 Sep 2022 07:21:36 GMT
Server: Apache/2.4.54 (Debian)
X-Powered-By: PHP/7.4.30
Vary: Accept-Encoding
Content-Length: 360
Content-Type: text/html; charset=UTF-8

<!doctype html>
<html lang="ja">
  <head>
    <meta charset="utf-8"/>
    <meta name="viewport" content="width=device-width,initial-scale=1"/>
    <title>post_receive</title>
  </head>
  <body>
    <h1>post receive</h1>
    <p>name is momotaro</p>
    <script>
    'use strict';

    </script>
  </body>
</html>
```

4.2 日本語文字列の場合

日本語文字列の場合は、やはり URL エンコードをする。以下は、桃太郎を URL エンコードして送信している。

```
POST /post_receive.php HTTP/1.1
Host: 192.168.31.27
Content-Length:32
Content-Type: application/x-www-form-urlencoded

name=%E6%A1%83%E5%A4%AA%E9%83%8E
```

以下がそのレスポンスである。y

```
HTTP/1.1 200 OK
Date: Sat, 10 Sep 2022 07:27:23 GMT
Server: Apache/2.4.54 (Debian)
X-Powered-By: PHP/7.4.30
Vary: Accept-Encoding
Content-Length: 361
Content-Type: text/html; charset=UTF-8

<!doctype html>
<html lang="ja">
  <head>
    <meta charset="utf-8"/>
    <meta name="viewport" content="width=device-width,initial-scale=1"/>
    <title>post_receive</title>
  </head>
  <body>
    <h1>post receive</h1>
    <p>name is 桃太郎</p>
    <script>
    'use strict';

    </script>
  </body>
</html>
```