WEBプログラミング/演習
第8回

スケジュール

- 第1回：Webプログラミングの基本
- 第2回：JavaScriptの基本
- 第3回：オブジェクト
- 第4回：関数
- 第5回：オブジェクト指向
- 第6回：クライアントサイドJavaScript
- 第7回：公開APIの利用(1)
- 第8回：公開APIの利用(2)
- 第9回：Pythonプログラミングの基礎(1)
- 第10回：　Pythonプログラミングの基礎(2)
- 第11回：サーバサイドプログラミング(1)
- 第12回：サーバサイドプログラミング(2)
- 第13回：サーバサイドプログラミング(3)
- 第14回：学習内容の振り返り

公開APIの利用（2）

CoursePower（第8回>第8回プログラム一式）からダウンロードください．これらはexercise8（フォルダ名）として扱いましょう．

今日の課題について

- 今日もGCPを【使います】
- まずは，exercise8のフォルダのファイルを，GCPにアップロードください.

- GCPが必須なわけではなく，Webサーバが必須になります.
  - ローカルで，webServer.pyを動かしても実施可能だとは思います.
  - この意味がわかる人のみローカルで実施して構いません.

今日の内容と復習

- 前回の課題4をベースにもう少し色々いじりながら，JavaScriptを全体的に復習する回にしたいと思います．

- 結局callback関数ってなんだったの？

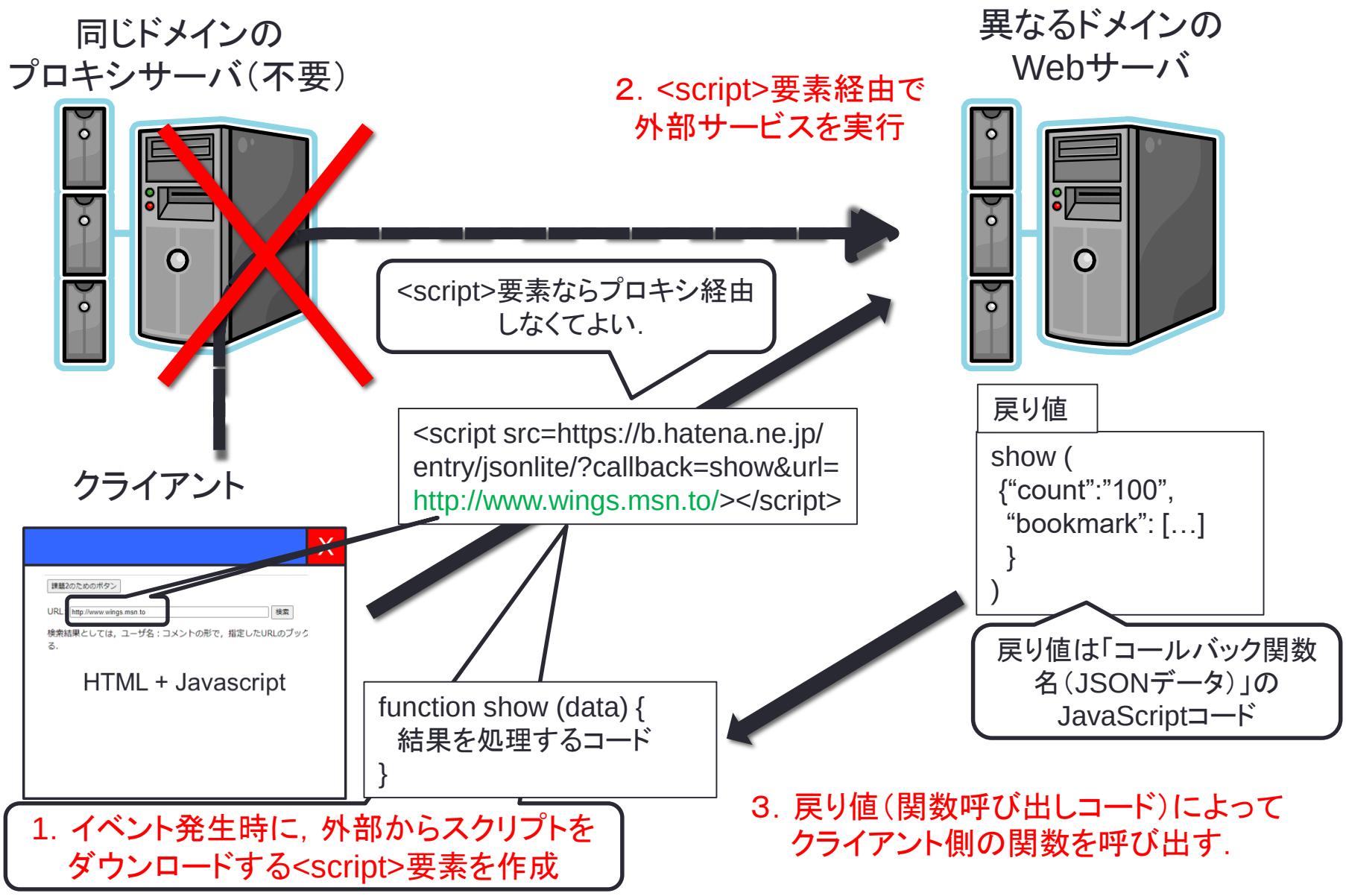
```
document.addEventListener('DOMContentLoaded', function () {
    let result = document.getElementById('result');

    document.getElementById('btn').addEventListener('click', function () {
        let target_url = document.getElementById('url').value; //HTML上で入力されたURLを受け取る
        let url = 'http://b.hatena.ne.jp/entry/jsonlite/?callback=show&url=' + encodeURIComponent(target_url);
        //callbackパラメータには，結果を処理する関数を指定する
        //urlパラメータには，ブックマーク情報を受け取りたいURLを指定する
        //urlはencodeURIComponentを使って，getのパラメータとして適切な文字になる
        let scr = document.createElement('script');
        scr.src = url; //ややこしいけど，scriptタグなので，scrって変数名，sorce
        document.getElementsByTagName('body').item(0).appendChild(scr);
    }, false);
}, false);
```

異なるドメインに直接アクセスするために，<script>要素を作っている．

callbackのパラメータで指定している関数がcallback関数

WebAPIの実行結果のデータが渡される関数になる

script要素を利用したcallback関数のイメージ



callback関数その2

- 具体的にどんな動きをしていたのか？

```
//==========課題4のためのコード==========
function show(data) {
    if (data === null) {
        result.textContent = 'ブックマークは存在しませんでした';
    } else {
        let bms = data.bookmarks;
        //返ってきたjsonはjavascriptのオブジェクトとして扱える
        //bookmarksラベルのデータ（配列）にアクセスしていると考えればよい
        let ul = document.createElement('ul');
        for (let i = 0, len = bms.length; i < len; i++) {
            let li = document.createElement('li');
            let anchor = document.createElement('a');
            anchor.href = 'http://b.hatena.ne.jp/' + bms[i].user;
            let text = document.createTextNode(bms[i].user + ' : ' + bms[i].comment);
            anchor.appendChild(text);
            li.appendChild(anchor);
            ul.appendChild(li);
```

引数 data は，WebAPIの実行結果のjsonデータがそのまま入ってくる

jsonデータは，JavaScriptではオブジェクト型（連想配列）で扱われる

前回の課題3で確認した構造と見比べながら確認しよう．
dataは右の構造のルートに当たると見ればよい．
ルートの中には，titleとscreenshotとcountとbookmarksがある．
そのbookmarks部分だけ取り出している．
これは，ブックマーク数分だけあるので，配列になっている．

```
{
    “title”: …,
    “screenshot”: …,
    “count”: …,
    “bookmarks”: [{
        “user”: …,
        “comment”: …,
        “timestamp”: …,
        “tags”: […]
    },{…},…],
}
```

callback関数その3


```
//==========課題4のためのコード==========
function show(data) {
    if (data === null) {
        result.textContent = 'ブックマークは存在しませんでした';
    } else {
        let bms = data.bookmarks;
        //返ってきたjsonはjavascriptのオブジェクトとして扱える
        //bookmarksラベルのデータ（配列）にアクセスしていると考えればよい
        let ul = document.createElement('ul');
        for (let i = 0, len = bms.length; i < len; i++) {
            let li = document.createElement('li');
            let anchor = document.createElement('a');
            anchor.href = 'http://b.hatena.ne.jp/' + bms[i].user;
            let text = document.createTextNode(bms[i].user + ':' + bms[i].comment);
            anchor.appendChild(text);
            li.appendChild(anchor);
            ul.appendChild(li);
}
```

bookmarks(=bms)の
配列の中身を取得したいので，
for文で回している

bookmarksの一つ一つには，user，comment，timestamp，tags が入っているので，その中の，userとcommentを取り出している．

課題4のためには，ここで，tagsを取得する必要があった．
tagsの中身は，配列になっており，タグ（キーワード）が区別されて格納されている．

```
{
    “title”: …,
    “screenshot”: …,
    “count”: …,
    “bookmarks”: [{
        “user”: …,
        “comment”: …,
        “timestamp”: …,
        “tags”: […]
    },{…},…],
}
```

課題4-1では何をしたらよかったのか？

- 基本的には元のshowを
  もとに次のように変更

まず，showTags関数を作る
（ここをshowTagsに書き換える）

この行はユーザ情報へのリンクを
作っているだけなので，
使わないので削除する

user，commentに変わり，tagsを
使いたいので，
bms[i].tags
に書き換える

最後に，callbackパラメータを
showTagsにしてあげる
（ここをshowTagsに書き換える）

```
//==========課題4のためのコード==========
function show(data) {
    if (data === null) {
        result.textContent = 'ブックマークは存在しませんでした';
    } else {
        let bms = data.bookmarks;
        //返ってきたjsonはjavascriptのオブジェクトとして扱える
        //bookmarksラベルのデータ（配列）にアクセスしていると考えればよい
        let ul = document.createElement('ul');
        for (let i = 0, len = bms.length; i < len; i++) {
            let li = document.createElement('li');
            let anchor = document.createElement('a');
            anchor.href = 'http://b.hatena.ne.jp/' + bms[i].user;
            let text = document.createTextNode(bms[i].user + ':' + bms[i].comment);
            anchor.appendChild(text);
            li.appendChild(anchor);
            ul.appendChild(li);
        }
        result.appendChild(ul);
        //どんな感じでHTMLにタグを追加しているのか想像しながら読もう
    }
}
```

```
document.addEventListener('DOMContentLoaded', function () {
    let result = document.getElementById('result');

    document.getElementById('btn').addEventListener('click', function (
        let target_url = document.getElementById('url').value; //HTML上で入力されたURLを受け取る
        let url = 'http://b.hatena.ne.jp/entry/jsonlite/?callback=show&url=' + encodeURIComponent(target_url);
        //callbackパラメータには，結果を処理する関数を指定する
        //urlパラメータには，ブックマーク情報を受け取りたいURLを指定する
```

Webページ側との対応

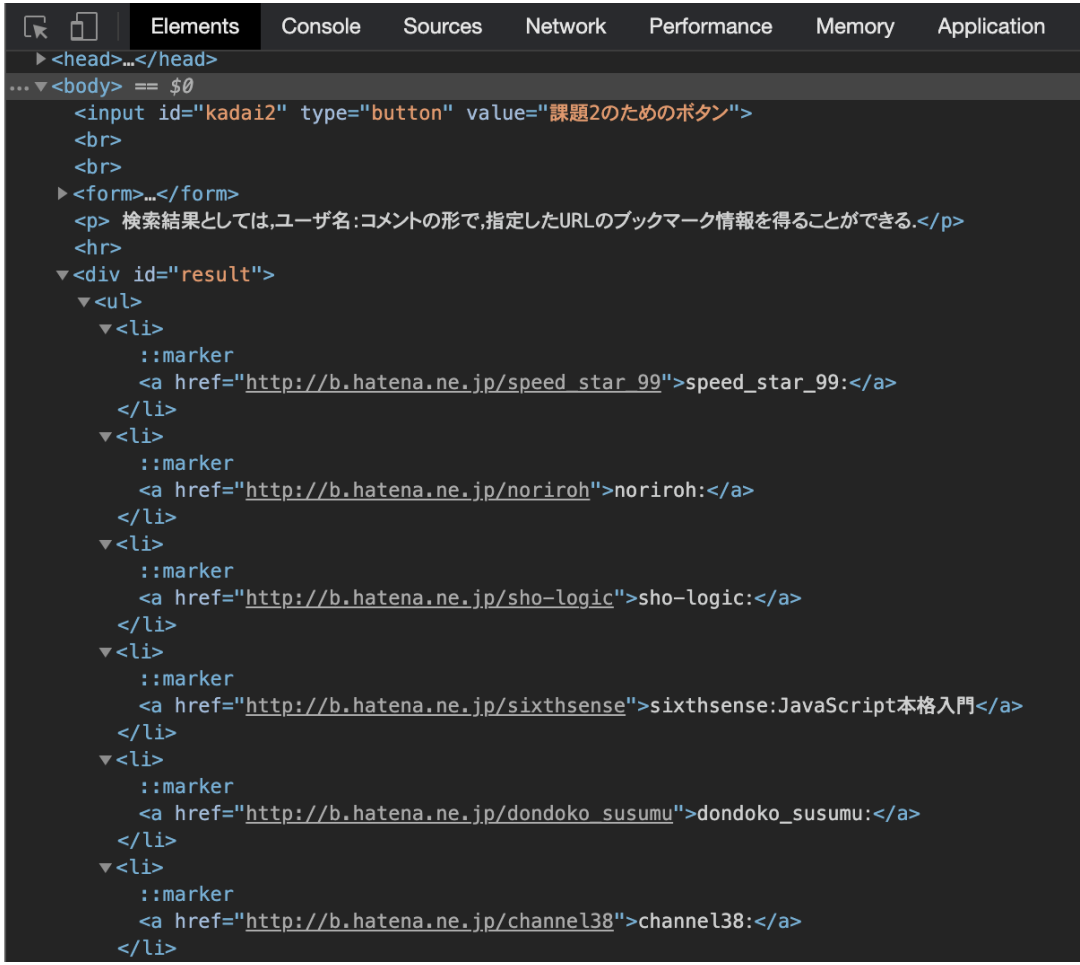
- 実行後の画面と生成されたHTML

課題2のためのボタン

URL: http://www.wings.msn.to 検索

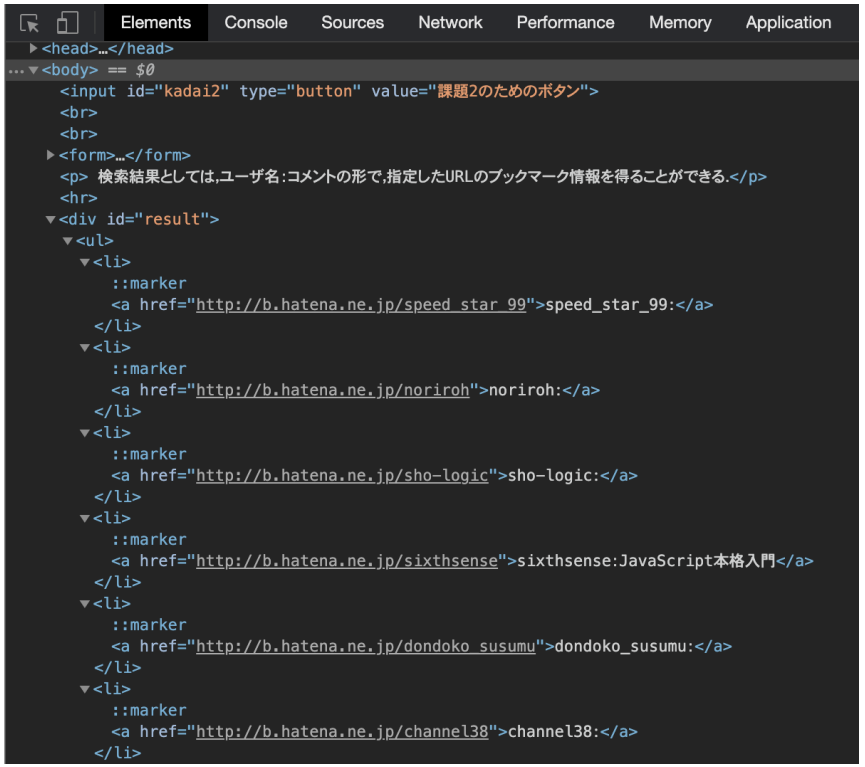
検索結果としては，ユーザ名：コメントの形で，指定したURLのブックマーク情報を得ることができる．

- speed_star_99：
- noriroh：
- sho-logic：
- sixthsense：JavaScript本格入門
- dondoko_susumu：
- channel38：
- nightrider_a0001：
- zazu0311：
- amashio：
- maaa328：
- yamashiro0110：
- ikosin：
- kovayan：
- nacika_inscatolare：本買ったよ
- As_hsp：本買ったよ



Webページ側との対応 その2


```
//==========課題4のためのコード==========
function show(data) {
    if (data === null) {
        result.textContent = 'ブックマークは存在しませんでした';
    } else {
        let bms = data.bookmarks;
        //返ってきたjsonはjavascriptのオブジェクトとして扱える
        //bookmarksラベルのデータ（配列）にアクセスしていると考えればよい
        let ul = document.createElement('ul');
        for (let i = 0, len = bms.length; i < len; i++) {
            let li = document.createElement('li');
            let anchor = document.createElement('a');
            anchor.href = 'http://b.hatena.ne.jp/' + bms[i].user;
            let text = document.createTextNode(bms[i].user + ':' + bms[i].comment);
            anchor.appendChild(text);
            li.appendChild(anchor);
            ul.appendChild(li);
        }
        result.appendChild(ul);
        //どんな感じでHTMLにタグを追加しているのか想像しながら読もう
    }
}
```



- show関数での生成の流れと，実行時に生成されたHTMLとを対応付けて理解しよう
  - createElementはタグで生成（ul,li,a,text_node）
  - appendChiledはタグの子要素としてタグを接続
    - ul <- li <- a <- text

前回の課題4-2のために必要だったこと

- 結局，tagsは単なる配列なので，やることは

1. 配列の中の要素を集計する
2. 集計したものを表示する

この2つだけ

- 1に関しては，純粋なJavaScriptの問題
- 2に関しては，JavaScriptによるHTML作成の問題
- あと，0番として，JSONの任意の該当箇所を取得できること，がある（課題4-1の内容）．

- ということで今日は，これらを分解して，類するものを演習しましょう.

[課題1] 配列の値を色々触ろう

- WebAPI_2.jsの課題1の箇所に定義された配列を引数にとり，以下の結果を得る関数を書こう.
  - HTMLファイルは，kadai1you.htmlを使うとよい.
  - 各結果は，console.logで確認すればよい

  - 課題1-1：最大値を返す関数 //100が正しい
  - 課題1-2：最小値を返す関数 //2が正しい
  - 課題1-3：中央値を返す関数 //35.5が正しい
  - 課題1-4：最頻値を返す関数 //40が正しい
  - 課題1-5：不偏分散を返す関数 //約524.35
  - 課題1-6：各値が何回出たかを数えて連想配列にして返す関数. できれば出現回数で降順にすること.
    - 確認用データ（降順）だと，40が9回，54が6回，30が6回，18が6回，，，となっていく.
    - 降順のしかたは，前回課題4.2countTags関数の解答例が参考になるかも.

[課題2] Webページに結果を表示しよう

- 以下の仕様を満たす，WebAPI_2.htmlを作成せよ.
  - 合計3つのボタンが表示されることになる

  - 課題2-1
    - ボタンが押されたら，最大値がWebページ中に表示される
    - ただし，値はWebAPI_2.js中に書かれた課題1の配列値（2行目の配列ね）を用いること．最大値は課題1で作成した関数を使う.
  - 課題2-2
    - ボタンが押されたら，最大値，最小値，中央値，最頻値，分散がそれぞれWebページ中に箇条書きで表示される
    - ただし，値はWebAPI_2.js中に書かれた課題1の配列値（2行目の配列ね）を用いること．課題1で作成した関数を使う.
    - 箇条書きには，HTMLタグのul，liを用いること
  - 課題2-3
    - ボタンが押されたら，各値が何回出たかを数えて，Webページ中にその結果が箇条書きで表示される
    - ただし，値はWebAPI_2.js中に書かれた課題1の配列値（2行目の配列ね）を用いること
    - 箇条書きには，HTMLタグのul，liを用いること

[課題3] 任意の値を処理できるようにしよう

- WebAPI_2.htmlに入力用のテキストボックスを追加し，以下のことができるようにせよ（次頁にちょっとしたヒントあり）
  - 課題3-1
    - 課題2-1のボタンが押されたときに， WebAPI_2.js中に書かれた課題1用配列値ではなく，入力用のテキストボックスに入力されたカンマ区切りの数値列（例：54, 40, 10, 5, 21, …）を用いるように変更せよ
  - 課題3-2
    - 課題2-2のボタンが押されたときに， WebAPI_2.js中に書かれた課題1用配列値ではなく，入力用のテキストボックスに入力されたカンマ区切りの数値列を用いるように変更せよ
  - 課題3-3
    - 課題2-3のボタンが押されたときに， WebAPI_2.js中に書かれた課題1用配列値ではなく，入力用のテキストボックスに入力されたカンマ区切りの数値列を用いるように変更せよ

[課題3] ヒント

- テキストボックスに入力された数値（列）は文字列として認識されているので以下の対処が必要である.

- まず，テキストボックス入力された数値列から数値だけを取り出す（＝カンマ区切りを外していく）.
  → split()を使って，数値列の前から順番に
  カンマを取り外しながら数値を取得していく.

- 取り出した数値は文字列扱いなので数値に変換する.
  → Number()を使って数値に変換するとよい.

書き方にもよるが，以上の処理を関数getInputData()とかにしておくと，課題3-1,3-2,3-3ごとに毎回書かなくてよくても便利かも.

成果物の提出（〆切：11/19中）

- 作成および編集した以下のファイルを提出せよ
  - WebAPI_2.js
  - WebAPI_2.html

- 提出はCoursePowerへ.