jQuery とは？

・2006 年リリースされた「JavaScript のライブラリ」
jQuery の中身は JavaScript で構成されたプログラム。

■ jQuery のメリット

❶. ブラウザを意識せずコーディングできる。
❷.DOM 操作とイベント処理を簡潔に記述できる。
❸. アニメーション処理を簡潔に記述できる。
❹.Ajax（非同期処理）を簡潔に記述できる。

？「ライブラリ」とは？

ある程度まとまったプログラムを
あらかじめ作ってくれているファイルのこと。

導入方法 ～html で必要な記述～

STEP.❶ jQuery ファイルの読み込み

jQuery を利用する為には、開発している HTML の中に jQuery を読み込む必要がある。

＜ファイルをダウンロードしてサーバーに直接アップする方法＞

```
<head>
  <script src="https://code.jquery.com/jquery-3.5.1.min.js"></script>
</head>
```

STEP.❷ JS ファイルにコードを記述する。

jQuery を書き始める場合は下記の様な構文を記述する（これは**<head>**内にコードを書く場合）

jQuery

```
$(function() {
  // ここに jQuery の処理を記述（実際の処理を書く場所）
});
```

処理自体を囲っている記述は「ページが読み込まれたとき」を表す構文です。
従来通り、</body>直前に記載する場合は、下記の基本書式をそのまま書けばOK。

jQuery の基本的な書き方

```
$(' セレクタ '). メソッド (' パラメータ [ 引数 ]');
```

jQueryオブジェクト

＜例＞ $('h1').css('color','red');　$('.items').fadeIn(3000);

？「$」とは？

jQuery で最も重要な関数。jQuery() と同じ関数。jQuery は、この jQuery 関数と、それによって取得される jQuery オブジェクト、jQuery オブジェクトが持つプロパティやメソッドが機能の中心となっています。

DOMの取得 / セレクタの書き方

$ 関数の引数に CSS セレクタを使用することで該当する DOM 要素の jQuery オブジェクトを取得することができる。

```
$('p') - 要素セレクタ
$('#wrapper') – id セレクタ
$('.button') – クラスセレクタ
$('li a') – 子孫セレクタ
$("p, h2") – グループセレクタ
```

javascript でも CSS セレクタから DOM 要素を取得することができる。
.querySelector('#wrapper');
.querySelector('.item');

jQuery メソッド

```
. メソッド (' パラメータ [ 引数 ]');
```

＜例＞$(".hoge").css("color","red");

メソッドはオブジェクトを操作する命令文を指します。

メソッドチェーン

```
$(' セレクタ '). メソッド (). メソッド (). メソッド ();
```

＜例＞$(".hoge").css("color","red").attr("title","hoge");

同じ要素に対して複数の命令（メソッド）を鎖（チェーン）のように繋いで記述する方法。

onメソッド

```
$(' セレクタ ').on('イベント名', function() {
        // イベント適用後に実行する処理
});
```

javascript でも CSS セレクタから DOM 要素を 取得することができる。
.addEventListener('click',() => {});

引数にイベント名と関数を渡すことで使用できます。
1 つ目の引数にイベント名を、2 つ目の引数にイベント発生時の処理をコールバック関数で指定します。

css メソッドの値を複数設定

```
$("p").css({
  color:"red",
  font-size:"16px",
  font-weight:"bold"
});
```

javascript でも style プロパティから操作できるが
複数同時に行うことはできない。
.style.backgroundColor;

同じような記述を書かなくて済む！

各jQueryメソッドについて

メソッド群① CSS,Attributes(属性)系のメソッド

CSS、Attributesのメソッドは、CSSや属性を取得、変更することができるメソッドです。
CSSに直憬記述しても良いが動的にCSSのスタイルを査更することができれば、Webサイトをもっと魅力的にするこ とができるはず。

メソッド	説明
css( )	CSSのスタイルを操作する $("#navi a").cse("color"); カラーを政得します。 $("#navi a").css("color", "red"); カラーをセットします。上記は#navi内にあるリンクの色を、赤にする.
width( )	幅属性を操作する $("a img").width(); 幅を取得します。 $("a img").width("150px"); 画像の幅に150pxをセットしています。
addClass( )	クラス属性を追加する $ ("navi a").addClass("current"); クラスを追加します. 例ではcurrentというクラスを付けています。
attr( )	属性を操作する (追加や取得など) $ ("a").attr("target","..blank"); アンカー (<a>タグ) にtarget属性を追加します。

メソッド群② Effects(エフェクト)系のメソッド

エフェクト系メソッドを使えば、 コンテンツをフェードさせて表示させたり、 スライドさせたりなど、HTML とCSSだけではな かなかできなかった動きをつける事が可熊！

メソッド	説明
toggle( )	要素の表示、非表示を切り替える $(".rudden").toggle(); 表示されてるものを隠したり、非表示のもを表示します。
fadeIn() **※fadeOut()**	要素をフェードイン (表示) する　　//要素をフェードアウト (非表示) する $(".fade").fadeIn(); フェードインさせて表示します。
show() **※hide()**	要素を表示する　　//要素を非表示にする $(".hidden").show(); 要素を表示します。
animate()	要素にアニメーションを実装する $("img").animate({ "height" : "0px" }); アニメーションしながら、高さを0にしています。

各jQueryメソッドについて

メソッド群③ Traversing(トラバース)系のメソッド

トラパース系メソッドは、H1'MLのDOMツリー内から指定条件に合うものを選択したり、検索するのに使います。 例えば以下のようなHTMLがあった場合、トラバース系のメソッドを使って、どういう事ができるのか見てみます。

メソッド	説明
find( )	要素の指定する $("#navi").find("li"); HTML内、#navi内のすべてのli指定する。
first() ※**last()**	最初の要素を指定する　　//最後の要素を指定する $ ("#navi li").first(); #navi liの中の最初のliを返します。
next()	次の要素を指定する $("#logo").next(); #logoの次の要素を返します。

メソッド群④ HTML、DOM ··· Manipulation(マニュビレーション)系のメソッド

HTML、DOM ··· Manipulation (マニュビレーション) 系のメソッドは 、HTMLに要素を追加したり、削除したりすることができます。
例えばスライダーのブラグインなどを使うと、自分ではHTMLに記述してないのに「次へ」「前へ」ボタンが表示される。あんな感じで、jQueryを使って自動的に璽素を作ったりするのに使います。

メソッド	説明
prepend()	要素を一番最初に追加する $("#navi ul").prepend(' <ahref="#'> Home ') ; 指定した要素の一番最初に、HTMLを追加します.
append()	要素を一番最後に追加する $("#navi ul").append(' Blog '); 指定した要素の一番最後に、HTMLを追加します.
after() ※**before()**	要素を後ろに追加する　　//要素を前に追加する $("#logo").after(' Web Design "); 指定した要素の後ろにHTMLを追加します。
wrap()	要素を囲んで追加する $("#header").wrap(' '); 帽定した要素を囲んでHTMLを追加します。
remove()	要素を削除する ·$("#navi").remove(); 指定した要素を削除します。

各jQueryメソッドについて

メソッド群⑤ Events（イベント）系のメソッド

【イベントの基本構文】
「on()」は、さまざまなイベント処理を記述するために使われるメソッドになります。
ところで、さまざまなイベント処理とはどのようなものでしょうか？例えば、次のようなイベントが考えられます。

・マウス操作（クリック、ホバー、移動など）が行われた時
・キーボードから何か入力された時
・フォームが送信された時
・任意のフォーム要素にフォーカスされた時
・画面がスクロールされた時
　etc...

このように、何らかのイベントが発生した時に特定の処理を行いたい場合に「on()」メソッドは活躍ます。

メソッド	説明
対象要素.on(イベント名, セレクタ, データ ,関数) 1 $('button').on('click', function() { 2 console.log('クリックされました！'); 3 })	
change	フォーム部品の状態に何らかの変化があった時に発動
click	要素がクリックされた時に発動
blur / focus	要素にフォーカスが当たったとき(focus)、外れたとき(blur)に発動
load	ドキュメントが読み込まれたあとに発動
resize	ウィンドウサイズが変化した時に発動
mouseup **※mousedown**	マウスのボタンが押された時(mousedown) ※離された時(mouseup)に発動
submit	フォームが送信された時に発動
error	何らかのJavaScriptエラーが発生した時に発動

```
<body>
<button>ボタン</button>

<script>
    $('button').on('click', function() {

        console.log('クリックされました！');

    })
</script>
</body>
```

各jQueryメソッドについて

複数イベントの扱い方

「on()」を使って複数のイベントを登録する方法について見ていきましょう。
例えば、クリック操作だけでなくボタン要素にマウスが近づいただけでもイベントを実行できるようにします。

```
<body>
    <button>ボタン</button>

    <script>
        $('button').on('click mouseenter', function() {
            console.log('クリックされました！');
        })
    </script>
</body>
```

「on()」メソッドの引数に注目してください。ボタン要素にマウスが侵入してきた時点でイベントを実行するには「mouseenter」を使います。そのため、「on('click mouseenter', 関数)」のように記述することで、2つのイベントを同時に実行することが出来るわけです。
これにより、マウスが近づいてもクリックしてもコンソールログにメッセージが出力されるようになります。

イベントにオブジェクトデータを渡す方法

「on()」メソッドの引数にはオブジェクト形式のデータを指定することも可能です。このデータは、関数に渡すことが可能なので任意のデータを活用した処理ができるようになるので便利です。

```
<body>
    <button id="one">ボタン１</button>
    <button id="two">ボタン２</button>

    <script>
        $('#one').on('click', {name: '太郎'}, showText);
        $('#two').on('click', {name: '花子'}, showText);

        function showText( e ) {
            console.log( e.data.name + 'さん、こんにちは！' );
        }
    </script>
</body>
```

実行結果

```
 ※「ボタン１」をクリックした例
太郎さん、こんにちは！

※「ボタン２」をクリックした例
花子さん、こんにちは！
```

この例では、2つのボタン要素を配置しています。それぞれのボタンごとにon()を使ってイベント処理を実装。
on()の引数にオブジェクト形式のデータを指定しており、それぞれ「name」の値に異なる名前を設定しています。
このデータにアクセスするには、「e.data.プロパティ名」のように記述すること実現します。
そこで、「e.data.name」と記述すれば、それぞれのボタンをクリックした時に設定されている名前が表示されるというわけです。

メソッド① 要素関連のメソッド(続)

説明	メソッド
.toggle()	要素の表示非表示を交互に切り替える
.get()	要素のオブジェクトを取得する
.prev()	指定した要素の直前にある兄弟要素を取得する
.prevAll()	指定した要素の前にある兄弟要素を全て取得する
.prevUntil()	指定した要素の直前の兄弟要素から指定した要素までの兄弟要素を取得する
.next()	指定要素の次の要素を取得する
.nextAll()	指定した要素の次以降にある兄弟要素を全て取得する
.nextUntil()	指定した要素の次以降にある兄弟要素から指定した要素までを取得する
.siblings()	指定した要素の兄弟要素を全て取得する
.closest()	要素から指定した要素の最も近い親要素を取得する
.contents()	テキストやコメントも対象とした全子要素を取得する
.addBack()	マッチした要素に加えて、１つ前にマッチした要素を加える

メソッド② 属性関連のメソッド

説明	メソッド
.val()	value の値を取得 / 設定する
.attr(属性名)	属性を取得する
.attr(属性名 , 値)	属性の値を取得する
.prop(プロパティ名)	プロパティ値を設定する
.prop(プロパティ名 , 値)	プロパティを取得する
.removeAttr(属性名)	属性を削除する
.removeProp(プロパティ名)	プロパティを削除する
.addClass()	class を追加する
.removeClass()	class を削除する
.toggleClass()	class の適用 / 非適用を切り替える
.css()	スタイルを取得する

参考サイト

https://javascript.programmer-reference.com/jquery-list-method/

→各メソッド名のリンクにサンプルがあるので動作確認使ってください。

メソッド③ ユーティリティ関連のメソッド（ユーティリティ）

メソッド	説明
$.merge()	配列を結合する
.select()	文字を全選択状態にする
.focus()	フォーカスを当てる
$.each()	要素に対して繰り返し処理を行う
.every()	配列の要素を順番にチェックする
.noop()	処理を行わない事を明示する
$.type()	型を判定して結果を文字列で取得する
$.get()	GET リクエストを送信する
$.post()	POST リクエストを送信する
$.getScript()	JavaScript ファイルを非同期で読み込む
$.parseXML()	XML 文字列を XML ドキュメントに変換する

メソッド④ ユーティリティ関連のメソッド（チェック関連）

メソッド	説明
$.isNumeric()	数値かどうか判定する
$.isArray()	配列かどうか判定する
$.isPlainObject()	オブジェクトかどうか判定する
$.isEmptyObject()	空かどうか判定する
$.isFunction()	関数かどうか判定する
$.isWindow()	window オブジェクトかどうか判定する

メソッド⑤ アニメーション関連関連のメソッド

メソッド	説明
.fadeTo()	フェード処理を行う
.slideUp()	スライドアップ処理を行う
.slideDown()	スライドダウン処理を行う
.slideToggle()	スライドアップ / スライドダウン処理を交互に行う