

# PHP/HTML講習会

---

合宿事前講座 JavaScript編 櫛田一樹

# もくじ

---

- ・ おさらい
- ・ 要素をグループ化しよう！
- ・ JavaScript…??
- ・ はじめよう！JSプログラミング！

# おさらい

---

- ・ CSSってなんだっけ？？  
→HTMLの見栄えを司る文書！
- ・ 絶対パスと相対パス
- ・ 文字コード  
→縁a縁ヨ縁?縁!!!

# 1: 要素をグループ化しよう！

---

今まででは、それぞれの要素をclassとかidを使って制御していた。

```
<p id="text">ここに何か文章が入ります</p>
```

# 1: 要素をグループ化しよう！

---

要素をグループ化してまとめて制御しよう！

# 1: 要素をグループ化しよう！

---

グループ化するタグは2つあります。

- **span**タグ : インライン化する
- **div**タグ : ブロック化する

# 1-1: インライン要素とブロック要素

インライン要素とブロック要素の違いを見てみよう！

```
1 <style>
2     .block {
3         background-color: blue;
4     }
5
6     .inline {
7         background-color: red;
8     }
9 </style>
10 <html>
11     <div class="block">こんにちは。</div>
12     <span class="inline">こんばんは。</span>
13 </html>
```

実行結果



こんにちは。

こんばんは。

**span(インライン)は中身のテキストの大きさ分だけ  
div(ブロック)は画面の横幅いっぱいに！！**

# 1-1: インライン要素とブロック要素

---

## インライン要素：

行の一部として扱われる。サイズ調整はできず、基本的に横に並ぶ

## ブロック要素：

レイアウトをひとつのまとまりとして扱う。

marginやpadding等で余白やサイズ調整が可能。

基本的に縦に並ぶが横に並べることも可能ではある。（「div 横並べ」とかで調べてね！）

## 2: Javascript!

---

「そもそもJavascriptてなんやねん」  
「Javaと何が違うねん」  
とか思ってないですか？

## 2: Javascript!

---

**JavascriptはJavaとは全く別で無関係！！**

(マリオカートと株式会社マリカーみたいなもの)

- Webページを構成するすべてのオブジェクトを操作できる！
- 世の中のWebページやPC, スマホアプリから最新モデルの車まで！  
いろいろなところで使われている！

# 3: はじめよう！JSプログラミング！

---

## JavaScriptスターターキット

- 3-1: Hello! JavaScript!** (テキストの出力)
- 3-2: JSに計算させてみよう！！** (変数)
- 3-3: イベントを完全理解しよう** (イベント)
- 3-4: 関数を使ってみよう！！** (関数)
- 3-5: 超簡単！FizzBuzz問題！！** (ループ, 条件分岐)
- 3-6: 配列をいじり倒そう！！** (配列)
- 3-7: 初心者卒業！DOM操作！！** (DOM操作)

# **3-1: Hello! JavaScript!**

---

はじめよう！JSプログラミング！

# 3-1: Hello! JavaScript!

こんな感じのHTMLコードを用意しよう！  
前回作ったCSSファイルとともに持ってくるといいかも？

The screenshot shows a code editor interface with a sidebar and a main editing area.

**Left Sidebar:**

- エクスプローラー (Explorer):
  - 開いているエディター (Open Editors): HelloJS.html (04回目)
  - HTML\_PHP:
    - 02回目
    - 03回目
    - 04回目 (selected)
    - CSS
    - HelloJS.html
  - 02回目.zip- アウトライン (Outline):
  - html
    - body
    - head
      - link
      - meta

**Main Area:**

HelloJS.html

```
04回目 > HelloJS.html > html
1  <!DOCTYPE html>
2  <html>
3
4  <head>
5      <meta charset="utf-8">
6      <title>HelloHTML</title>
7      <link rel="stylesheet" href=".//CSS/style.css">
8  </head>
9
10 <body>
11
12
13 </body>
14
15 </html>
```

# 3-1: Hello! JavaScript!

---

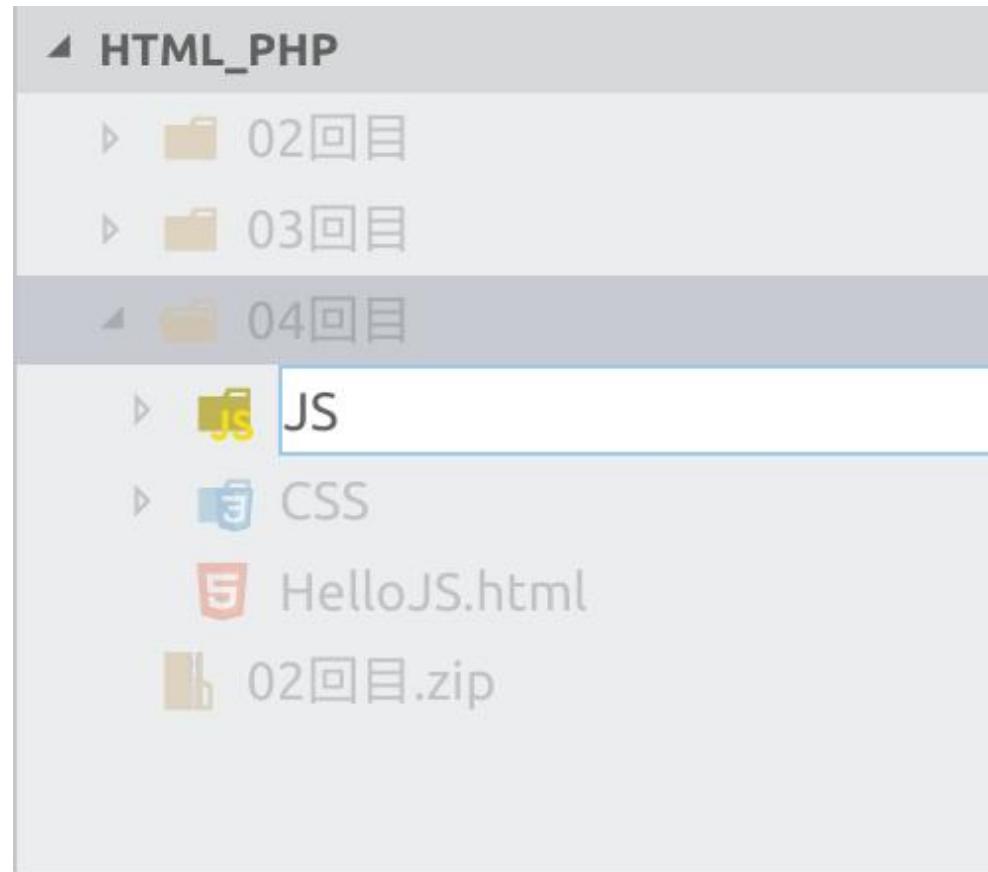
JavaScriptの書き方は2通りあるよ！

- ・HTML内の<script>タグ内に直接プログラムを書き込む方法
- ・外部にJSファイルを作ってそこから読み込む方法

今回はファイル整理のために後者の方法でいくよ！

# 3-1: Hello! JavaScript!

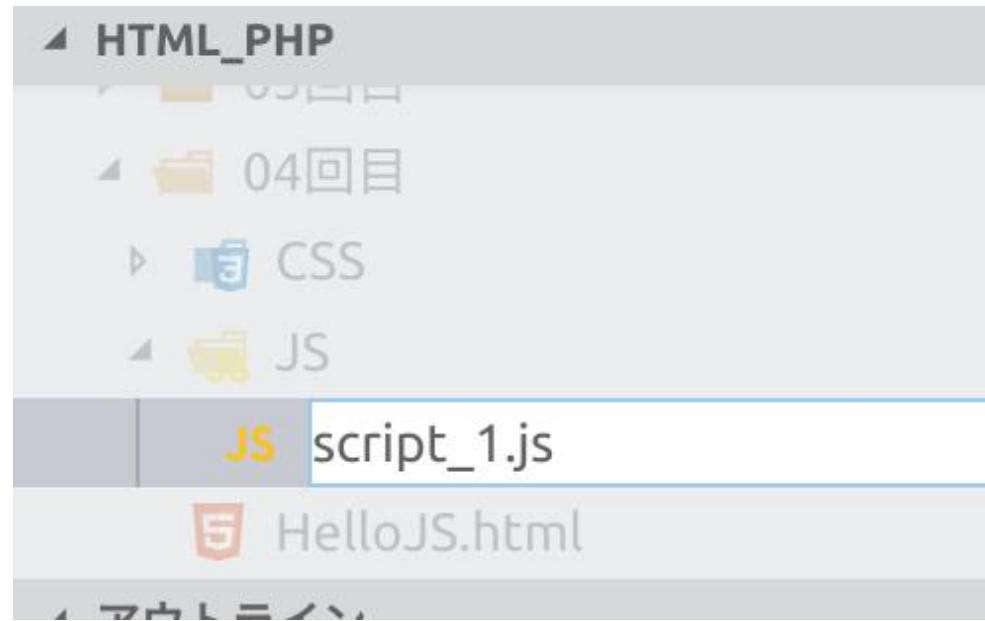
---



04回目のディレクトリの中に  
「JS」ディレクトリを作ろう！！  
(名前は何でもいいけどわかりやすい名前にしよう)

# 3-1: Hello! JavaScript!

---



JSの中に「script\_1.js」を作ろう！  
→ぶっちゃけ名前は何でもいい

# 3-1: Hello! JavaScript!

---

この一文を書こう！！

(C言語でいうとprintfみたいなもの！Webページに出力してくれるよ！)

04回目 ▶ JS ▶ **JS** script\_1.js

```
1   document.write("Hello, JavaScript!!");
```

# 3-1: Hello! JavaScript!

---

HelloJS.htmlに戻って<script>タグを書こう！

type要素の"text/javascript"で これがJSであると宣言するよ！



The screenshot shows a code editor interface with two tabs: "HelloJS.html" and "script\_1.js". The "HelloJS.html" tab is active, displaying the following code:

```
04回目 > HelloJS.html > html
1   <LINK rel="stylesheet" href="./CSS/style.css">
2
3
4
5
6
7
8   </head>
9
10  <script type="text/javascript" src="JS/script_1.js"></script>
11
12  <body>
13
```

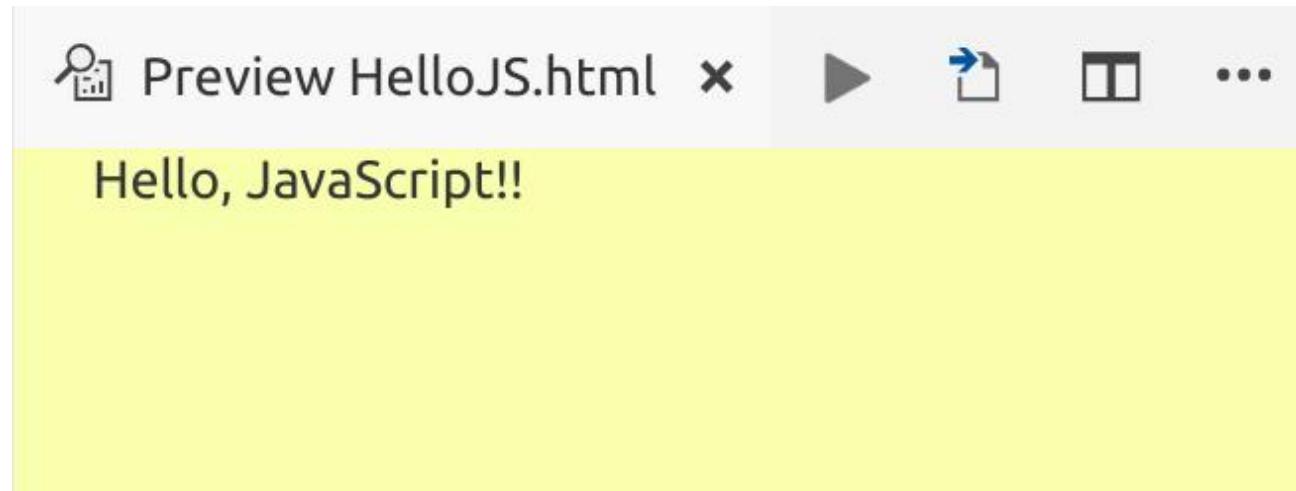
The "script\_1.js" tab is visible in the background. The code editor has a light gray background with syntax highlighting for HTML tags (blue) and CSS/JavaScript (orange). Line numbers are shown on the left.

# 3-1: Hello! JavaScript!

---

ブラウザで確認しよう！！

こんな感じに「Hello, JavaScript!!」と表示されていれば成功！！



## **3-2: JSに計算させてみよう！！**

---

はじめよう！JSプログラミング！

# 3-2: JSに計算させてみよう！！

---

まあ楽勝だと思いますがせっかくなので計算させてみましょう。

JSでの変数の作り方とかがわかる人は読み飛ばしてもらって結構です

$$7+5=12$$

$$7-5=2$$

$$7*5=35$$

$$7/5=1.4$$

$$7\%5=2$$

# 3-2: JSに計算させてみよう！！

calc.htmlを新しく作って、helloJS.htmlの内容をコピペしよう！

script\_2.jsをJSディレクトリ内に作ってHTML内で読み込もう！



```
calc.html × JS script_2.js
04回目 ▶ calc.html ▶ html
6 <title>HelloHTML</title>
7 <link rel="stylesheet" href=".//CSS/style.css">
8 </head>
9
10 <script type="text/javascript" src="JS/script_2.js">
11 </script>
12
```

# 3-2: JSに計算させてみよう！！

```
calc.html      JS script_2.js ×  
04回目 ▷ JS ▷ JS script_2.js ▷ ...  
1  var data1 = 7;  
2  var data2 = 5;  
3  
4  var data3 = data1 + data2;  
5  document.write(data1 + "+" + data2 + "=" + data3 + "<BR>"); // 加算(足し算)  
6  
7  data3 = data1 - data2;  
8  document.write(data1 + "-" + data2 + "=" + data3 + "<BR>"); // 減算(引き算)  
9  
10 data3 = data1 * data2;  
11 document.write(data1 + "*" + data2 + "=" + data3 + "<BR>"); // 乗算(掛け算)  
12  
13 data3 = data1 / data2;  
14 document.write(data1 + "/" + data2 + "=" + data3 + "<BR>"); // 除算(割り算)  
15  
16 data3 = data1 % data2;  
17 document.write(data1 + "%" + data2 + "=" + data3 + "<BR>"); // 余剰(あまり)
```

script.jsの中身を書いていこう！

varで変数を宣言するよ！  
(文字も数字もみんなvarに入れられる)

→**型推論**といいます。ぼくは嫌いです。  
型推論で調べてみてください。

ちなみに後から学ぶPHPも型推論です。

# 3-2: JSに計算させてみよう！！

---

ブラウザで実行させてみよう！！

こんな感じに出てくれば成功です！！

$$7+5=12$$

$$7-5=2$$

$$7*5=35$$

$$7/5=1.4$$

$$7\%5=2$$

## 3-2-1: 変数をひとつ減らしてみよう

---

実は変数を減らすやり方が存在します！  
一緒にやっていきましょう！！

## 3-2-1: 変数をひとつ減らしてみよう

---

data3とかいう変数、消せそうじゃないですか？？

document.write内で計算させてしまえば……???

```
4 var data3 = data1 + data2;  
5 document.write(data1 + "+" + data2 + "=" + data3 + "<BR>"); // 加算(足し算)  
6
```

# 3-2-1: 変数をひとつ減らしてみよう

---

こんな感じに変更してみた！！

data3のあった場所で変数どうしの計算をさせているよ！

このとき**括弧をつけないと変な感じになったりする**ので注意！

```
4 document.write(data1 + "+" + data2 + "=" + (data1+data2) + "<BR>"); // 加算(足し算)
5 document.write(data1 + "-" + data2 + "=" + (data1-data2) + "<BR>"); // 減算(引き算)
6 document.write(data1 + "*" + data2 + "=" + (data1*data2) + "<BR>"); // 乗算(掛け算)
7 document.write(data1 + "/" + data2 + "=" + (data1/data2) + "<BR>"); // 除算(割り算)
8 document.write(data1 + "%" + data2 + "=" + (data1%data2) + "<BR>"); // 余剰(あまり)
```

## 3-2-1：変数をひとつ減らしてみよう

---

ブラウザで実行させて同じように表示されたら大成功！！

$$7+5=12$$

$$7-5=2$$

$$7*5=35$$

$$7/5=1.4$$

$$7\%5=2$$

## **3-3: イベントを完全理解しよう**

---

はじめよう！JSプログラミング！

# 3-3: イベントを完全理解しよう

---

ボタンをクリックしても何もないとななんか寂しい……  
何かアクションをさせたいなあ……

event.htmlを作ってcalc.htmlの中身をコピペしよう

# 3-3: イベントを完全理解しよう

event.htmlのbody部に以下を書き込もう！



The screenshot shows a code editor interface with two tabs: "event.html" and "script\_3.js". The "event.html" tab is active, displaying the following HTML code:

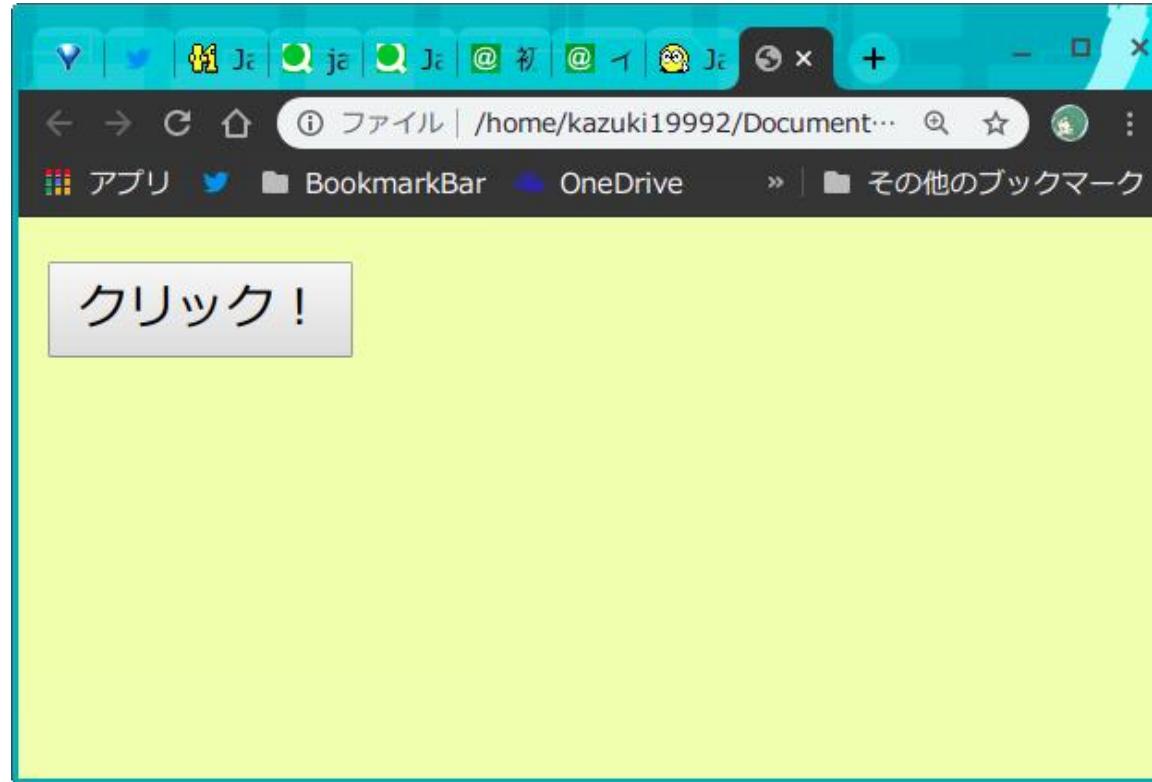
```
04回目 > event.html > html
11 </script>
12
13 <body>
14   <input type="button" value="クリック！" onclick="alert('うんち！w')">
15 </body>
16
17 </html>
```

The code consists of a single button element with an onclick event handler that triggers an alert dialog with the message 'うんち！w'.

# 3-3: イベントを完全理解しよう

---

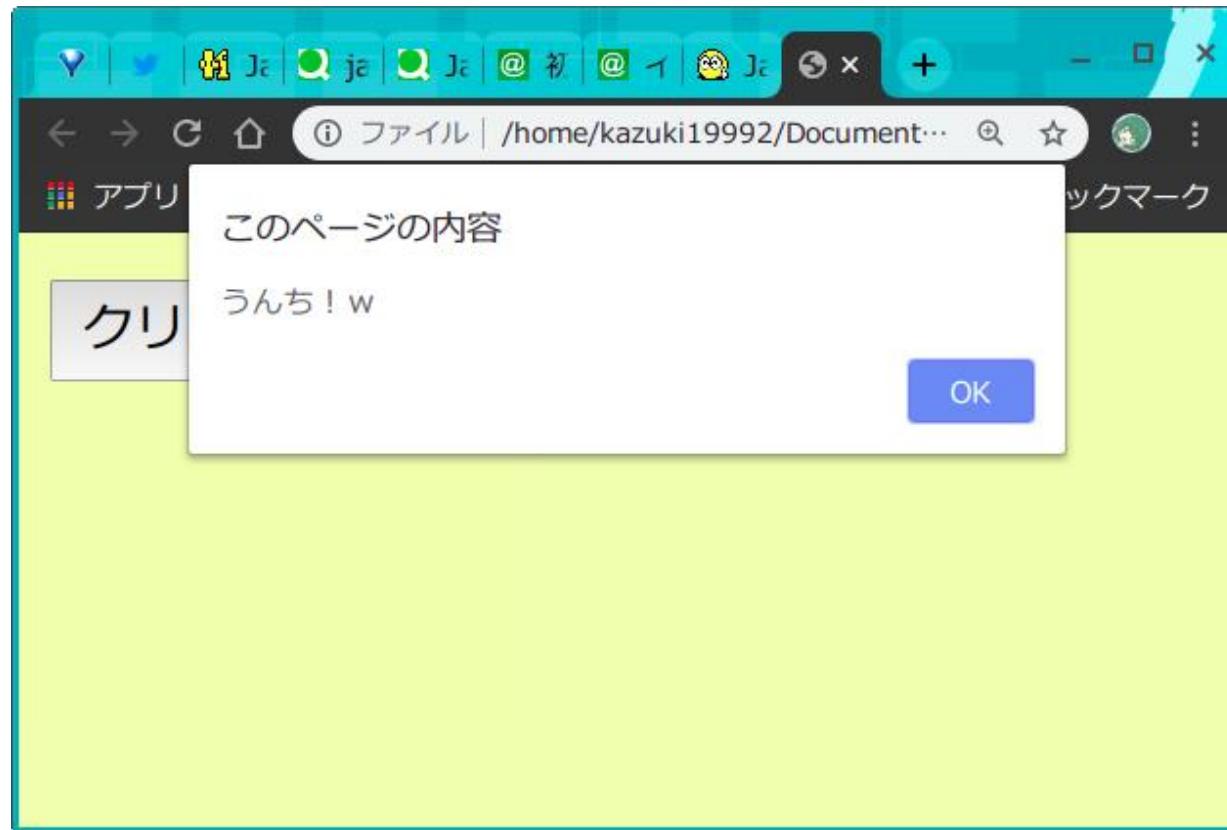
ブラウザで開こう！！  
「クリック！」と書かれたボタンが現れたね！



# 3-3: イベントを完全理解しよう

---

ボタンをクリックしてみよう！  
alert()が出てきたね！



## 3-3: イベントを完全理解しよう

---

「クリックされた」ということをきっかけにプログラムが動作した

このきっかけのことを 「**イベント**」 という！

# 3-3: イベントを完全理解しよう

---

このHTMLでは onclick以降がJavaScriptになります

- onclick : オブジェクトをクリックした時
- alart() : 警告ウインドウを表示する

今回はonclickで" "が使用されているから  
alart内では' 'を使っているよ！

onclickとalartで同じものを使うと正常に動作しないよ！

```
<body>
  <input type="button" value="クリック！" onclick="alert('うんち！w')">
</body>
```

## 3-3-1: いろいろなイベント

---

JSには様々なイベントがあります。他にもあるから調べてみてね!

- **onmouseover** : マウスポインタが重なったら
- **onmouseout** : マウスポインタが外れたら
- **onmousedown** : ボタンを押した時
- **onmouseup** : ボタンを離した時
- **ondblclick** : オブジェクトをダブルクリックした時
- **onload** : Webページが読み込まれた時
- **onunload** : Webページが閉じられた時
- **oncontextmenu** : 右クリックメニューが開かれようとした時

# 3-3-2: ボタンをイベントで操作しよう！

イベントでボタンの文字を変えてみよう！



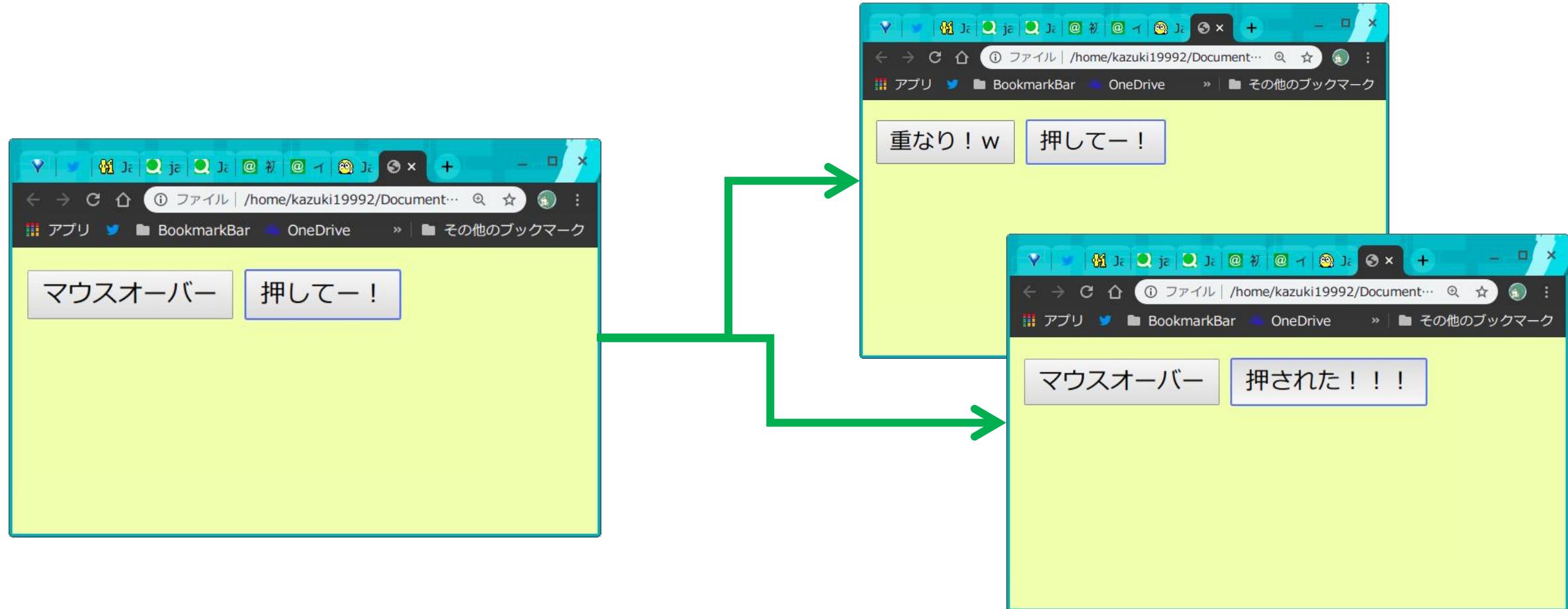
The screenshot shows a code editor interface with two tabs: "event.html" and "script\_3.js". The "event.html" tab is active, displaying the following HTML code:

```
04回目 > event.html > html
11 </script>
12
13 <body>
14     <!-- マウスオーバーしたときだけボタンの文字が変わる -->
15     <input type="button" value="マウスオーバー" onmouseover="this.value='重なり！w'" onmouseout="this.value='マウスオーバー'">
16
17     <!-- ボタンを押したときだけ文字が変わる -->
18     <input type="button" value="押してー！" onmousedown="this.value='押された！！！'" onmouseup="this.value='押してー！'">
19 </body>
20
```

The code uses inline JavaScript to change the button's value based on mouse events (mouseover,mouseout) and button events (mousedown,mouseup).

# 3-3-2: ボタンをイベントで操作しよう！

ブラウザで開いてみると…？？



# 3-3-3: ページを開いたら……??

---

ページを開いたら警告が出るようにしてみよう！！

bodyタグに直接書いてみよう



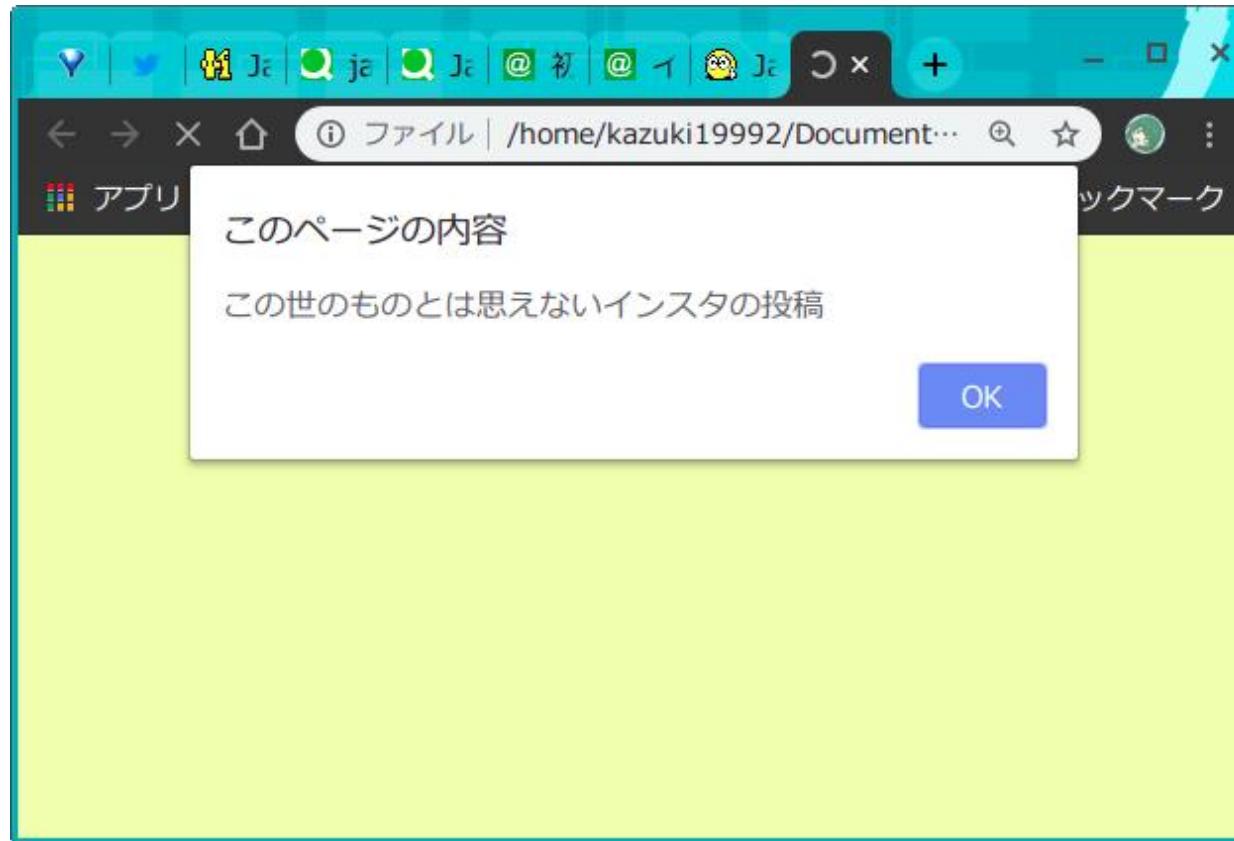
The screenshot shows a code editor interface with two tabs: "event.html" and "script\_2.js". The "event.html" tab is active, showing the following code:

```
04回目 > event.html > html
11 </script>
12
13 <body onload="alert('この世のものとは思えないインスタの投稿')">
14     <!-- マウスオーバーしたときだけボタンの文字が変わる -->
15     <input type="button" value="マウスオーバー" onmouseover="this.value='マウスオーバー'; this.value='マウスオーバー'" />
```

# 3-3-3: ページを開いたら……??

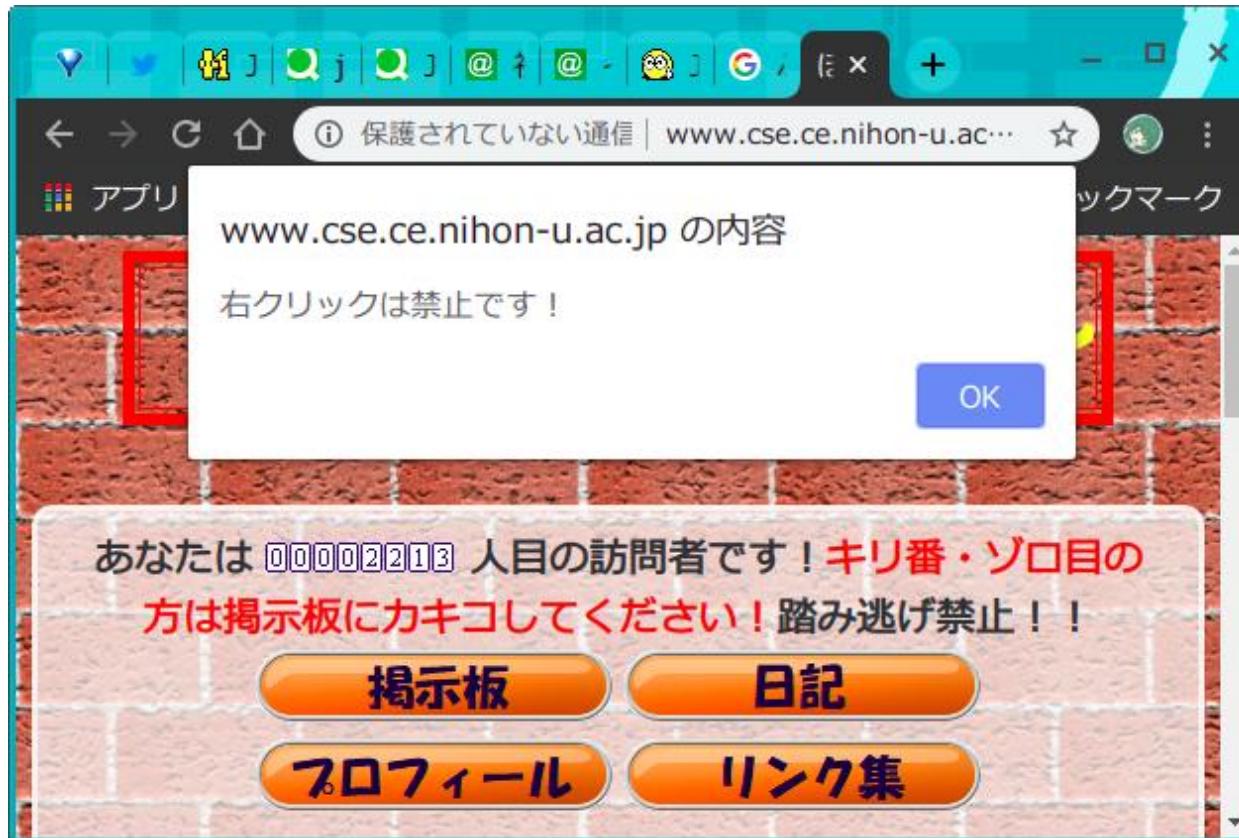
---

ブラウザでページを開いてみよう！！



# 3-3-4: 右クリックは禁止です！w

ぼく初心者だしソースコード見られたくないよお:(ノ`\_ゝノ)؛  
右クリック禁止して見られないようにしたいなあ……



←こんな感じに

# 3-3-4: 右クリックは禁止です！ w

bodyタグをこのように変更しよう！

return false というのは、操作を無かったことにするものだよ！  
(警告を表示した後、右クリックを無かったことにする)

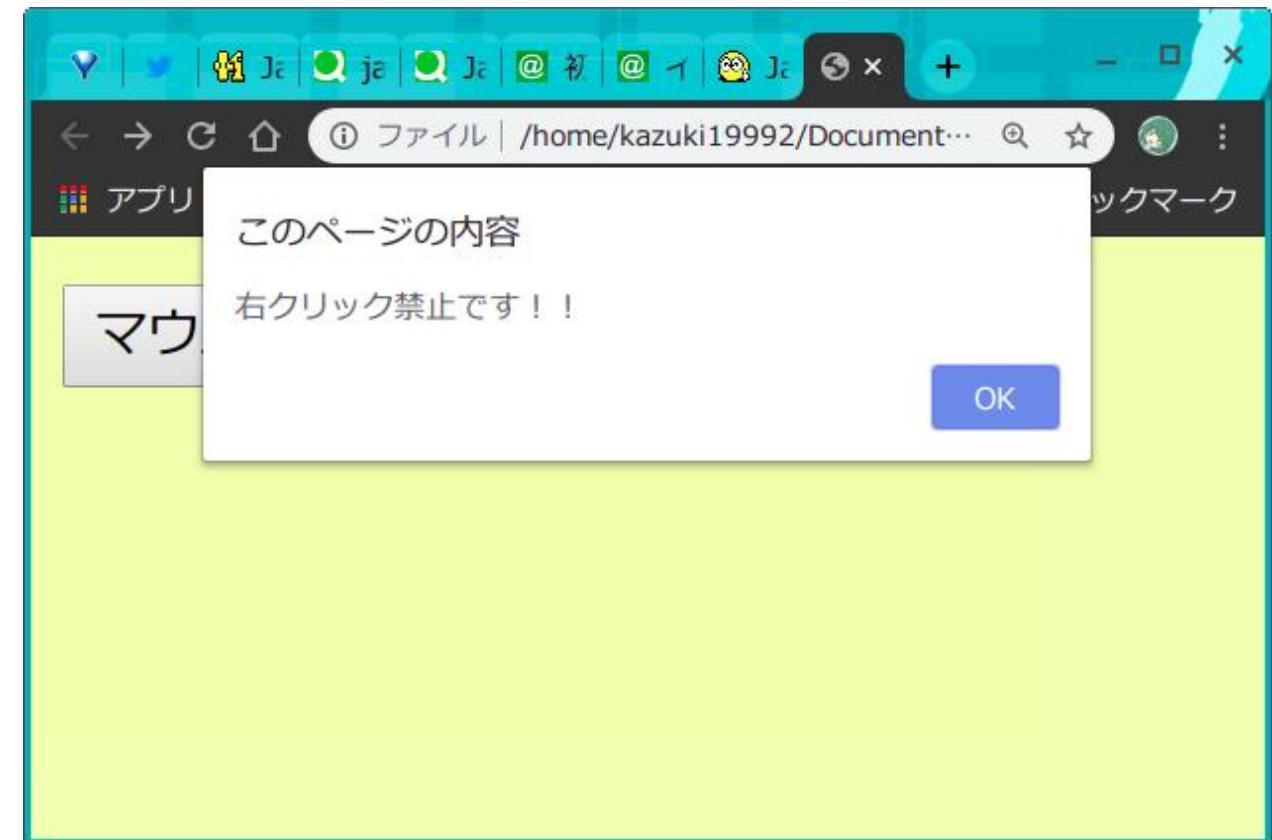
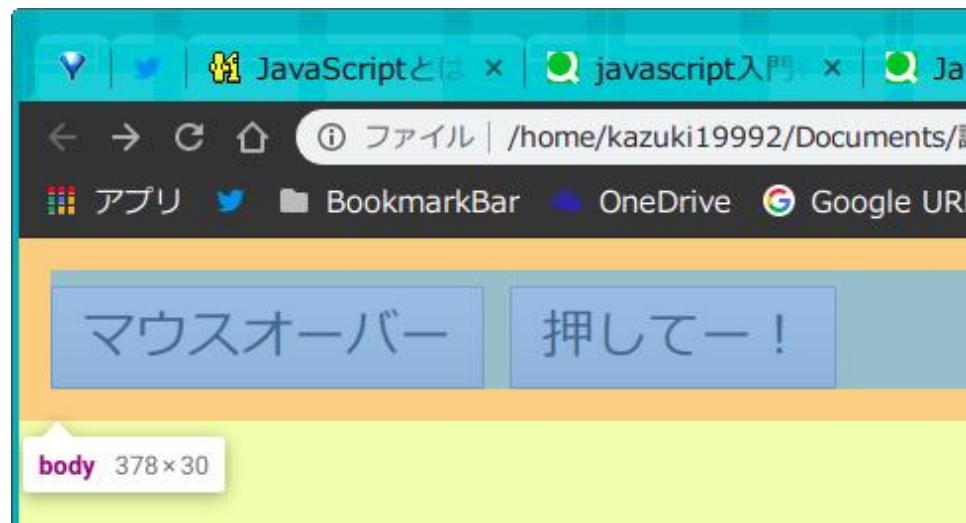


The screenshot shows a code editor interface with two tabs: "event.html" and "script\_2.js". The "event.html" tab is active. Below the tabs, there's a breadcrumb navigation: "04回目 > event.html > html". The main code area contains the following HTML and JavaScript:

```
9
10 <script type="text/javascript" src="JS/script_3.js">
11 </script>
12
13 <body oncontextmenu="alert('右クリック禁止です！！'); return false">
14     <!-- マウスオーバーしたときだけボタンの文字が変わる --&gt;
15     &lt;input type="button" value="マウスオーバー" onmouseover="this.val
16
17     &lt;!-- ボタンを押したときだけ文字が変わる --&gt;</pre>
```

# 3-3-4: 右クリックは禁止です！w

保存してブラウザで開いてみよう！！  
body部(左画像のオレンジ部分以内)を右クリックすると……



## **3-4: 関数を使ってみよう！！**

---

はじめよう！JSプログラミング！

# 3-4: 関数を使ってみよう！！

---

いろんな処理をやってみたい……  
関数ってのにも手を出してみるか！ｗ

function.htmlを作って、calc.htmlから中身をコピペしよう！  
script\_4.jsも忘れずに作ろう！

# 3-4: 関数を使ってみよう！！

---

まずは簡単なものから。

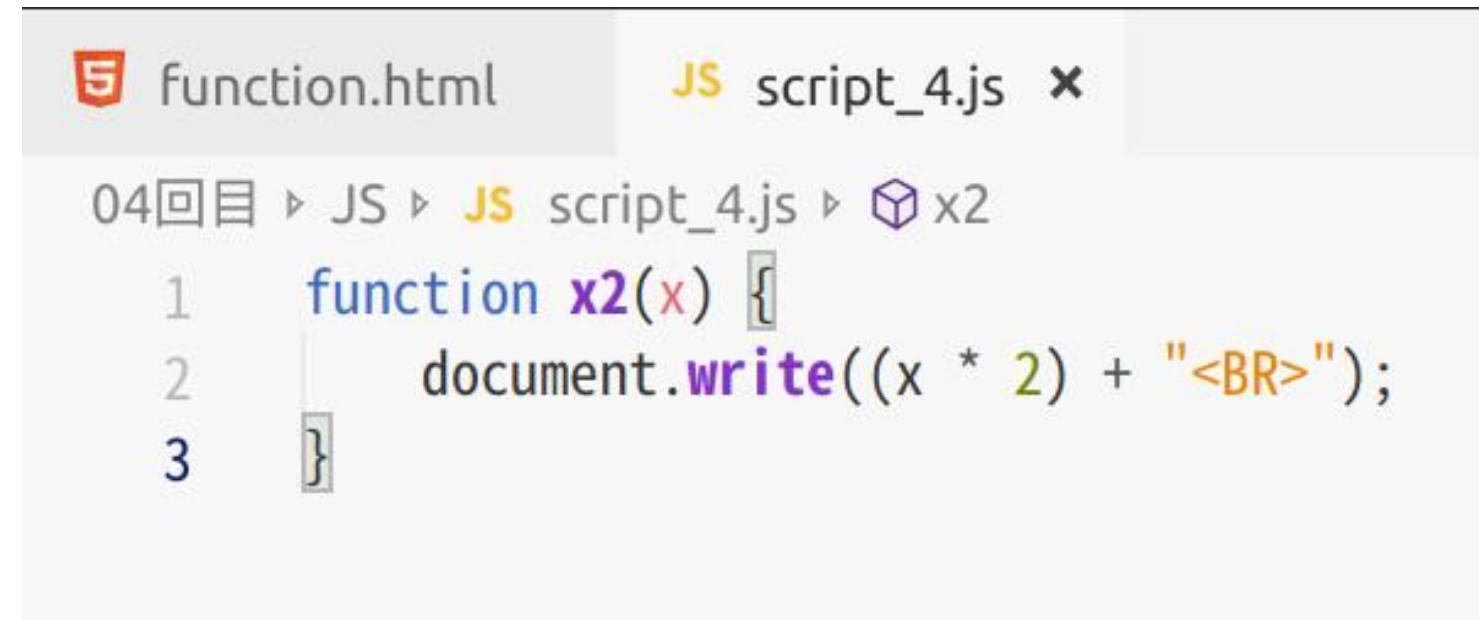
入力値を二倍にして出力する関数を作ろう！

# 3-4: 関数を使ってみよう！！

---

script\_4.jsを編集しよう！

関数名はシンプルに x2 にしよう！ xが入力値になるよ！  
(わかりやすければ何でもいいです)



The screenshot shows a code editor interface with two tabs: "function.html" and "script\_4.js". The "script\_4.js" tab is active, showing the following code:

```
04回目 ▶ JS ▶ JS script_4.js ▶ x2
1 function x2(x) {
2   document.write((x * 2) + "<BR>");
3 }
```

The code defines a function named x2 that takes a parameter x and writes the result of x multiplied by 2 plus a line break to the document.

# 3-4: 関数を使ってみよう！！

---

function.htmlを編集するよ！！

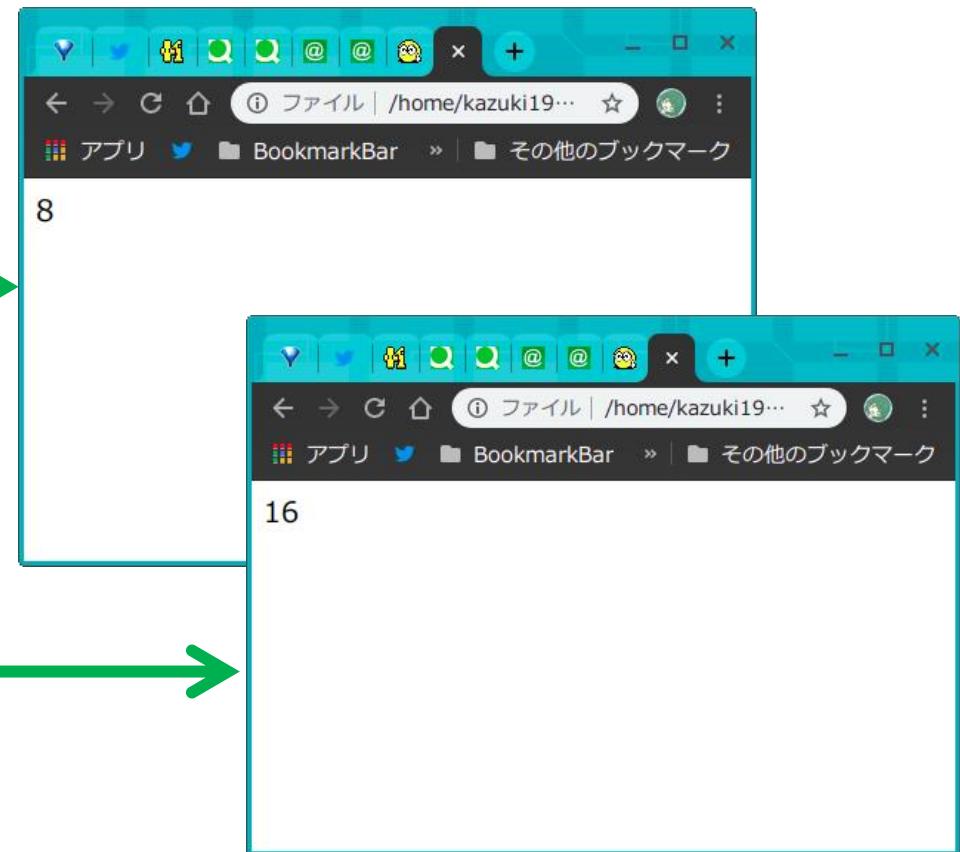
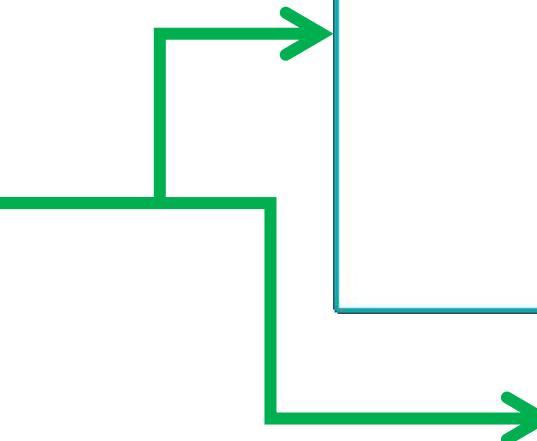
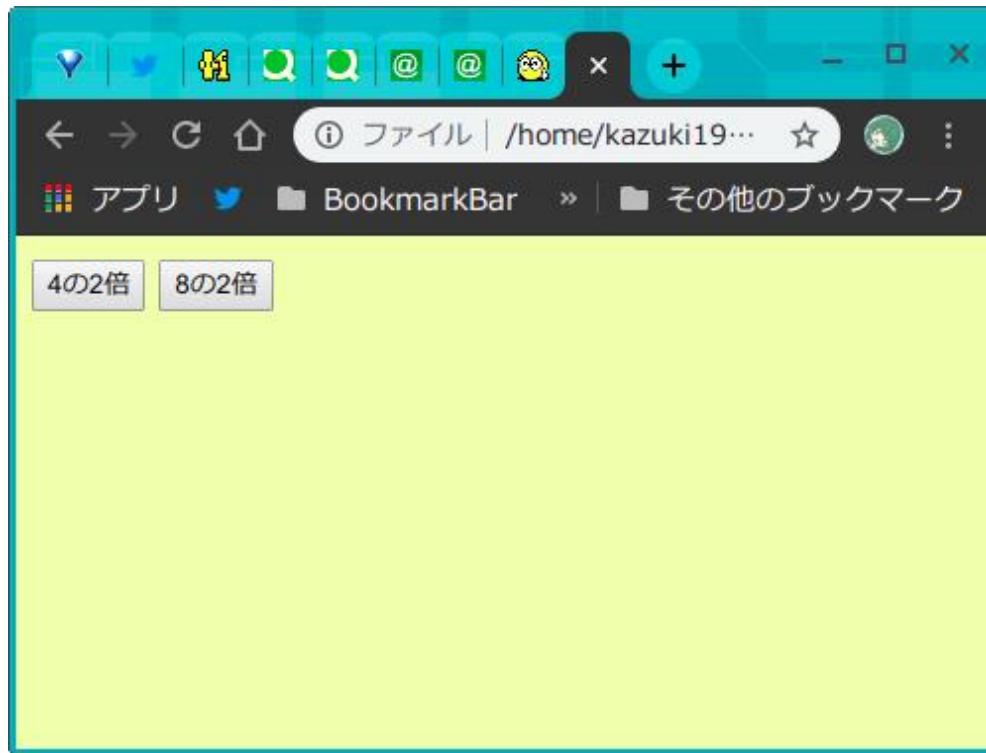
今回はボタンがクリックされたら関数が実行されるようにしよう！

```
10 <script type="text/javascript" src="JS/script_4.js">
11 </script>
12
13 <body>
14     <!-- クリックすると4の2倍の数字が出力される -->
15     <input type="button" value="4の2倍" onclick="x2(4)">
16
17     <!-- クリックすると8の2倍の数字が出力される -->
18     <input type="button" value="8の2倍" onclick="x2(8)">
19 </body>
20
21 □□□□□
```

# 3-4: 関数を使ってみよう！！

ブラウザで実行してみよう！！！

再読み込むと元の画面に戻ります。



# 3-4-1: 2倍 / 2乗計算機を作つてみよう！

---

はい。クソ簡単です。

入力された値を単に2倍にしたり2乗にする計算機です。

出力はalart()を使いましょう！

# 3-4-1: 2倍 / 2乗計算機を作つてみよう！

script\_4.jsの関数x2を以下のように編集しよう！

これと同じように関数squareを作ろう！！



The screenshot shows a code editor interface with the following details:

- File Structure:** 04回目 > JS > script\_4.js > square
- Code Snippet:**

```
1 function x2(x) {  
2     return (x * 2);  
3 }
```

# 3-4-1: 2倍 / 2乗計算機を作ってみよう！

```
function.html x JS script_4.js
04回目 > function.html > html > body
1  <!DOCTYPE html>
2  <html>
3
4  <head>
5      <meta charset="utf-8">
6      <title>HelloHTML</title>
7      <link rel="stylesheet" href="./CSS/style.css">
8  </head>
9
10 <script type="text/javascript" src="JS/script_4.js">
11 </script>
12
13 <body>
14     <!-- テキストボックス(値を入力させる) -->
15     <input type="text" id="num">
16     <br>
17
18     <!-- クリックすると入力値の二倍の数字が出力される -->
19     <input type="button" value="2倍" onclick="alert(x2(num.value))">
20
21     <!-- クリックすると入力値の二乗の数字が出力される -->
22     <input type="button" value="2乗" onclick="alert(square(num.value))">
23 </body>
24
25 </html>
```

イベント処理を使って実装するよ！！

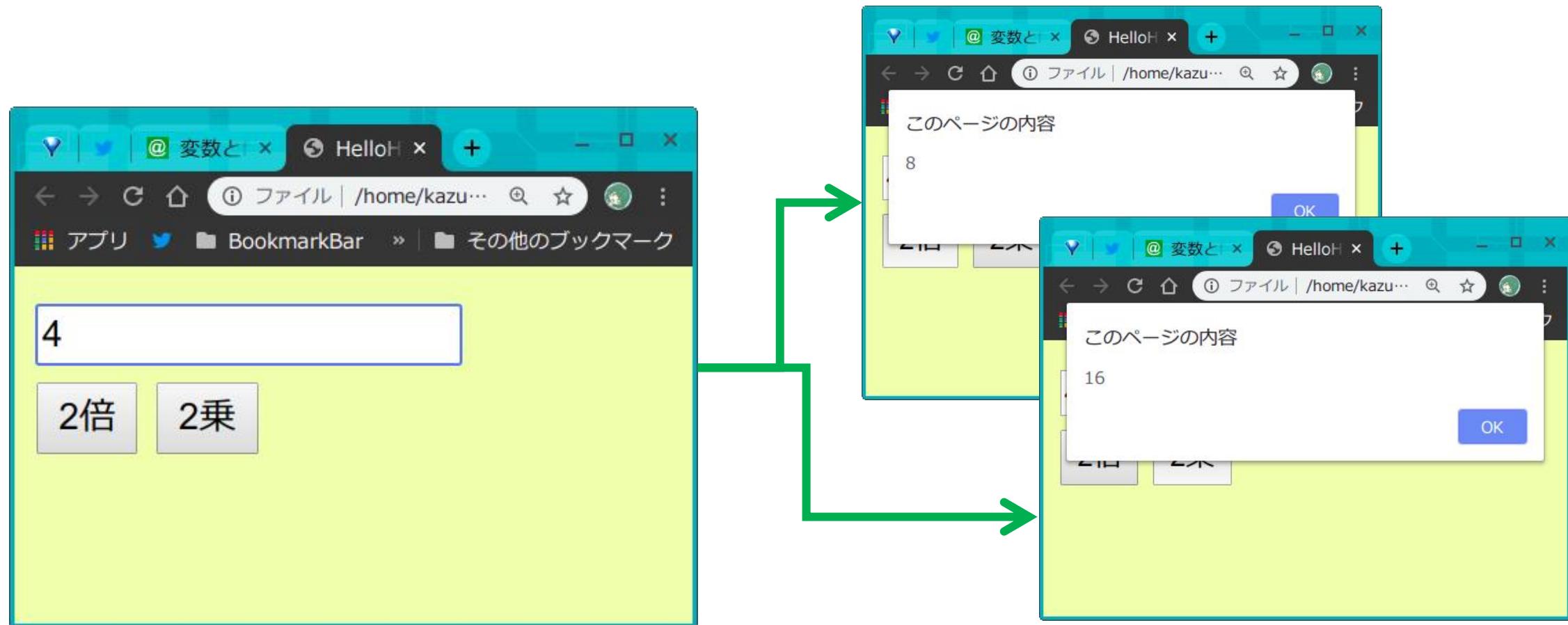
左のコードを書こう！！

num.valueでnumに入力されている  
値を渡すよ！！

alert内で関数x2とsquareが  
実行されているよ！

# 3-4-1: 2倍 / 2乗計算機を作ってみよう！

保存してブラウザで見てみよう！！



# 3-5: 超簡単！FizzBuzz問題！！

---

はじめよう！JSプログラミング！

# 3-5: 超簡単！FizzBuzz問題！！

---

最近はなんかよくわかんない人たちが  
「人工知能だ！」とか「AI AI AI AI !!!」と騒いでいます。  
彼らにとっては機械が何か判断を下せばそれはAIのようです。

せっかくなので  
**if文で超簡単にAI(笑)を作ってみましょう**

今から作るのはAIではないです(それはそう)  
条件分岐の練習用プログラムです。

# 3-5-1：プログラムに判断してもらおう！

---

入力されたデータが**奇数か、偶数か、それとも0か**  
判断してもらいましょう。

if.htmlを作り、中身をfunction.htmlからコピペしましょう。

# 3-5-1：プログラムに判断してもらおう！

body部をこのように記述しよう！



The screenshot shows a code editor window with the following details:

- File tab: if.html (highlighted)
- File tab: script\_5.js
- Path bar: 04回目 > if.html > html
- Code area:

```
9
10  <script type="text/javascript" src="JS/script_5.js">
11  </script>
12
13  <body>
14      <!-- テキストボックス(初期値を0にしておく) -->
15      <input type="text" value="0" id="num">
16      <BR>
17
18      <!-- クリックすると判断する関数が動く -->
19      <input type="button" value="ジャッジ" onclick="judge(num.value)">
20
21      <!-- 結果を出力させるエリア -->
22      <div id="result">ここに結果が出てくるよ！！</div>
23
24  </body>
25
```

# 3-5-1：プログラムに判断してもらおう！

script\_5.jsをこのように記述しよう！

```
if.html      JS script_5.js ×
04回目 ▶ JS ▶ JS script_5.js ▶ judge
1  function judge(input) {
2    if (input == 0) {
3      // inputが0だったら
4      document.getElementById("result").innerHTML = "入力は0やで！！";
5    } else if (input % 2 == 0) {
6      // inputが偶数だったら
7      document.getElementById("result").innerHTML = "入力は偶数やで！！";
8    } else if (input % 2 == 1) {
9      // inputが奇数だったら
10     document.getElementById("result").innerHTML = "入力は奇数やで！！";
11   } else {
12     // それ以外の場合(正常に値が入力されていなかった場合とか)
13     document.getElementById("result").innerHTML = "ちゃんと入力してや！！";
14   }
15 }
```

document.getElementById(" ").innerHTML

この関数は()の要素を  
置き換えてくれるもの！！

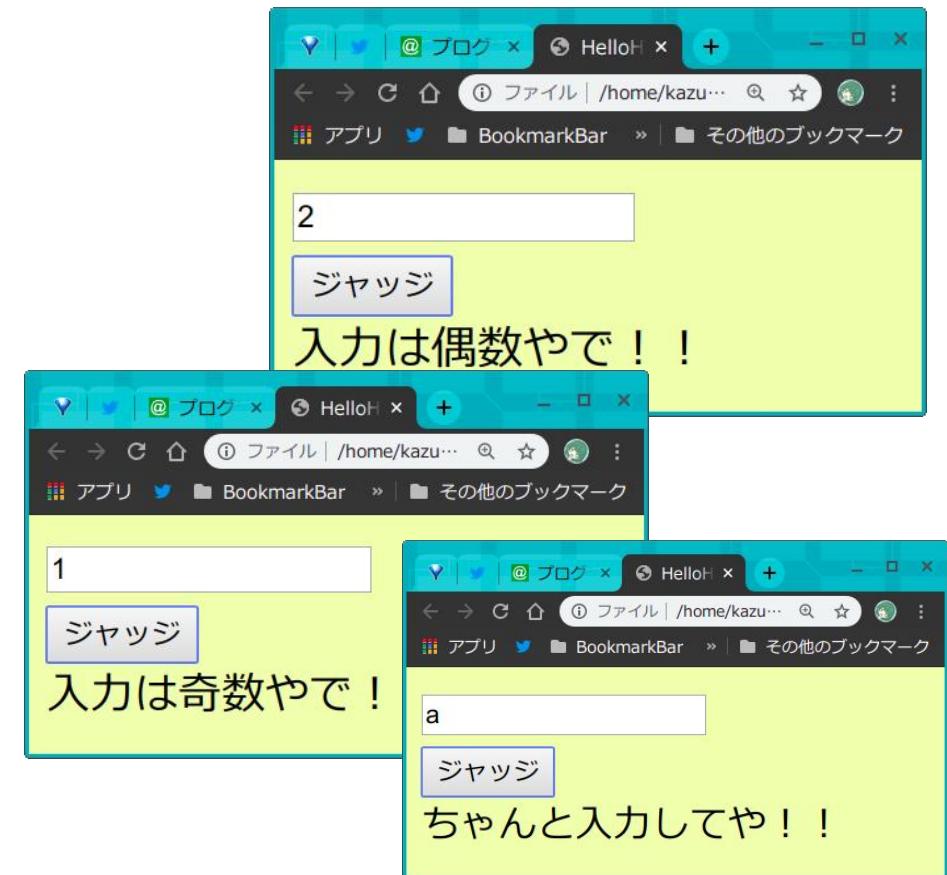
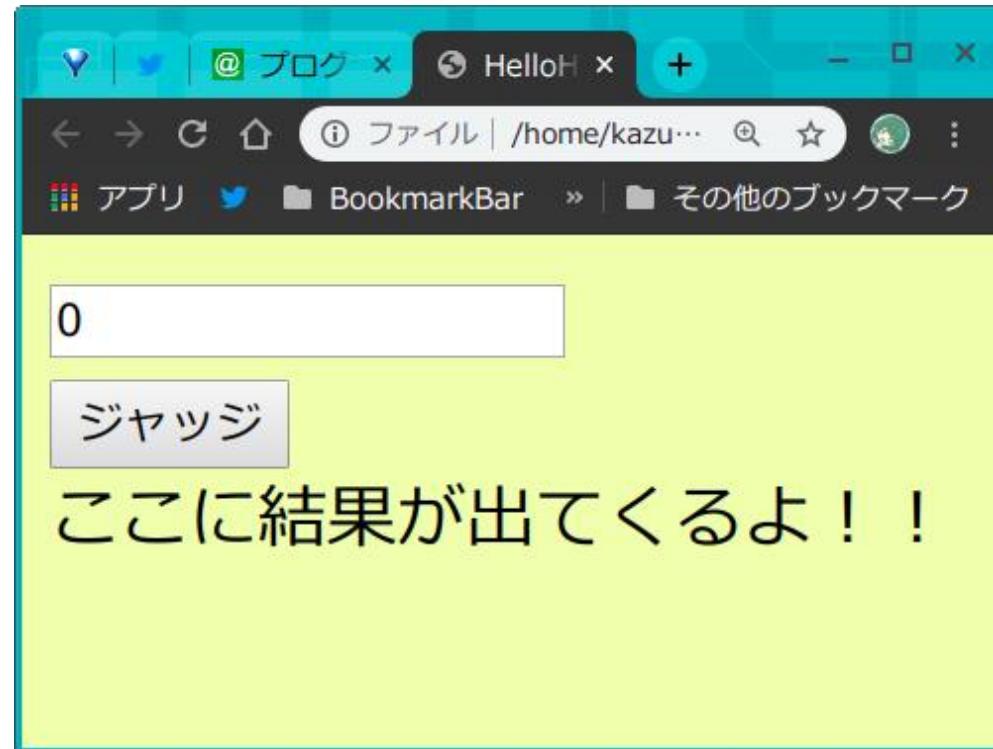
()は要素名(id指定)を入れよう！

いろいろ使いどころさんがあるから  
使ってみるといいかも！

基本的なif文の使い方はC言語と同じ！！  
わからない人はSlackで聞いてね！

# 3-5-1：プログラムに判断してもらおう！

保存してブラウザで見てみよう！！  
こんな感じに表示されれば成功だ！！



## 3-5-2: エンドレスエイト

---

繰り返し処理を実装してみよう！

これも基本的にはC言語と同じだよ！！

色々調べてみよう！

- **for文**
- **while文**
- **do while文 ←これくらい**

などなど……

# 3-5: 超簡単！FizzBuzz問題！！

---

ループの回数を指定して

- 3の倍数時に Fizz
- 5の倍数時に Buzz
- 15の倍数時に FizzBuzz

と出力しよう！！

# 3-5: 超簡単！FizzBuzz問題！！

---

fizzbuzz.htmlを新しく作ろう！！  
JSはscript\_5.jsに追記していくよ！

fizzbuzz.htmlはif.htmlからコピペしてね！

# 3-5: 超簡単！FizzBuzz問題！！

body部はここまで大きく変更はしないよ！

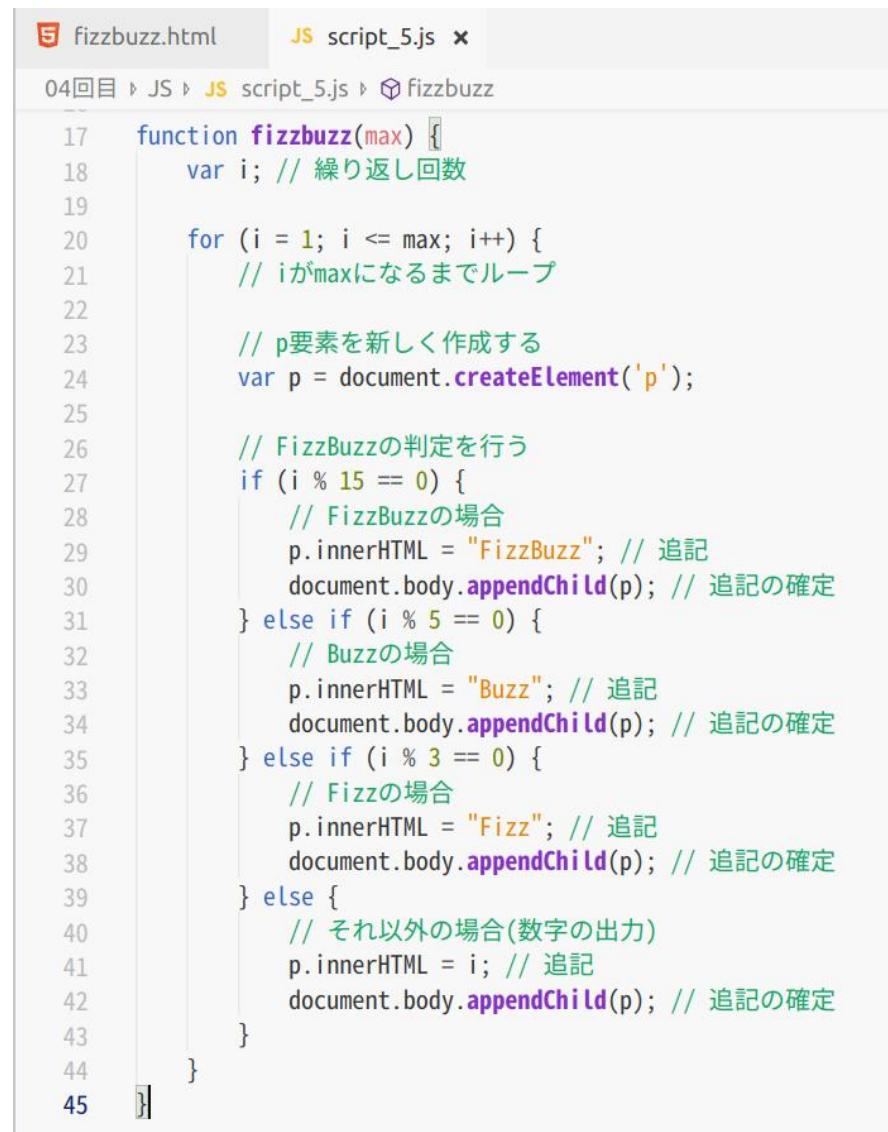


The screenshot shows a code editor interface with two tabs: "fizzbuzz.html" and "script\_5.js". The "fizzbuzz.html" tab is active, displaying the following HTML code:

```
13 <body>
14   <!-- ループ回数指定テキストボックス(初期値を0にしておく) -->
15   <input type="text" value="0" id="num">
16   <BR>
17
18   <!-- クリックがFizzBuzzのトリガーとなる -->
19   <input type="button" value="FizzBuzz開始" onclick="fizzbuzz(num.value)">
20
21 </body>
```

The code includes comments in Japanese explaining the purpose of each section: specifying the loop count via a text input, and defining a button that triggers the FizzBuzz logic.

# 3-5: 超簡単！FizzBuzz問題！！



The screenshot shows a code editor with two tabs: "fizzbuzz.html" and "script\_5.js". The "script\_5.js" tab is active, showing the following JavaScript code:

```
04回目 › JS › JS script_5.js › fizzbuzz
17 function fizzbuzz(max) {
18     var i; // 繰り返し回数
19
20     for (i = 1; i <= max; i++) {
21         // iがmaxになるまでループ
22
23         // p要素を新しく作成する
24         var p = document.createElement('p');
25
26         // FizzBuzzの判定を行う
27         if (i % 15 == 0) {
28             // FizzBuzzの場合
29             p.innerHTML = "FizzBuzz"; // 追記
30             document.body.appendChild(p); // 追記の確定
31         } else if (i % 5 == 0) {
32             // Buzzの場合
33             p.innerHTML = "Buzz"; // 追記
34             document.body.appendChild(p); // 追記の確定
35         } else if (i % 3 == 0) {
36             // Fizzの場合
37             p.innerHTML = "Fizz"; // 追記
38             document.body.appendChild(p); // 追記の確定
39         } else {
40             // それ以外の場合(数字の出力)
41             p.innerHTML = i; // 追記
42             document.body.appendChild(p); // 追記の確定
43         }
44     }
45 }
```

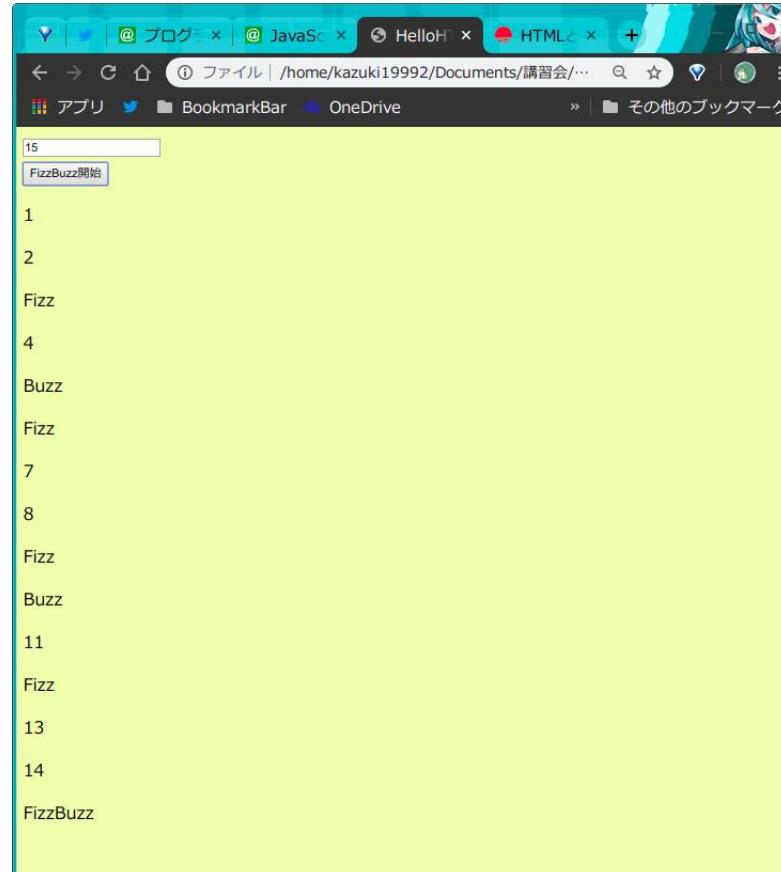
script\_5.jsは追記していくう！

新しい関数はコメント文を参照してね！

# 3-5: 超簡単！FizzBuzz問題！！

---

ブラウザで開いて適当な数字を入力していい感じに表示されたら成功！



# 3-6: 配列をいじり倒そう！！

---

はじめよう！JSプログラミング！

# 3-6: 配列をいじり倒そう！！

---

JavaScriptでは文字でも、数字でも何でもvarに入るというのは  
記憶に新しいと思います。やばいよね。

実はこのvar、これで配列も宣言できます。やばば。  
一緒にやってみましょう。

# 3-6: 配列をいじり倒そう！！

---

配列の宣言は2種類あります。  
同じ結果が得られるので好きな方を使いましょう。

- **var a = [];**  
→C言語での宣言と似ているようでちょっと違う。紛らわしい。
- **var a = new Array();**  
→これに至ってはC言語と全然違う。でもちょっとわかりやすい…？

# 3-6: 配列をいじり倒そう！！

配列に初期値を与える方法は2種類あります。  
基本的にC言語と同じです。

```
1 var array = new Array();  
2 array[0] = 0;  
3 array[1] = 3;  
4 array[2] = 2;  
5 array[3] = 5;
```

方法1：宣言した後にそれぞれ初期化する

もちろんこれでも大丈夫！

```
array.html      JS script_6.js ●  
1回目 ▶ JS ▶ JS script_6.js ▶ array  
1 var array = [0,3,2,5];  
2
```

```
1 var array = new Array(0,3,2,5);  
2
```

方法2：宣言と同時に初期化する

# 3-6: 配列をいじり倒そう！！

---

配列の制御方法も基本的にC言語と変わらないです！  
わからない人はSlackか合宿当日に聞きに来てね！

配列以外にもデータをまとめめる方法として  
**「オブジェクト」**とかいうものがあります。  
気になるようなら調べてみてね！

# **3-7：初心者卒業！DOM操作！！**

---

はじめよう！JSプログラミング！

# 3-7: 初心者卒業！DOM操作！！

---

よくここまでたどり着きました。  
お疲れ様でした！！

これをクリアすれば晴れて初心者卒業です。

よくここまで資料作った！！お疲れ様自分！！

# 3-7: 初心者卒業！DOM操作！！

---

DOM操作と一緒に勉強する前に、HTMLの基本構造について  
復習していきましょう。

これから学ぶDOM操作はHTMLの構造を理解していないとできないのです。  
文句言わずに一緒に頑張りましょ…？

# 3-7-1: HTMLの基本構造

---

04回目 › 5 キホン.html › html

```
1  <!DOCTYPE html>
2  <html>
3      <head>
4          <meta charset="utf-8">
5          <title>たいとる</title>
6      </head>
7      <body>
8          <div id="block1">ブロック1</div>
9          <div id="block2">ブロック2</div>
10     </body>
11 </html>
```

このHTMLを見てください。

HTMLは、<BR>や<img>以外の  
基本的なタグはみんな

**<タグ>で始まり**

**</タグ>で終わる**

という感じになっています

# 3-7-1: HTMLの基本構造

---

04回目 › 5 キホン.html › html

```
1  <!DOCTYPE html>
2  <html>
3      <head>
4          <meta charset="utf-8">
5          <title>たいとる</title>
6      </head>
7      <body>
8          <div id="block1">ブロック1</div>
9          <div id="block2">ブロック2</div>
10     </body>
11 </html>
```

で、よく見てみると、

<html>の中に<head>と<body>があり、

<head>の中に<title>とかがあり……

<body>の中に<div>とかがある……

という感じになっています。

# 3-7-1: HTMLの基本構造

---

結局のところ何が言いたいかというと……

**HTMLは入れ子構造によってできている！！**

ということなんです！

この入れ子構造を「階層構造」と言ったりもするよ～

最近なところでは「OSのフォルダ」とかも階層構造だよ！  
→フォルダの中にフォルダがあるように、タグの中にタグがある

# 3-7-1: HTMLの基本構造

---

で、これが理解できるかどうかで  
DOM操作が理解できるかどうかが変わってきます。

わからない人はもう一度HTMLをお勉強したほうがいいかも……  
Slackで聞いてくれれば教えるよ！

# 3-7: 初心者卒業！DOM操作！！

---

さて、ここまでこれた優秀な皆さん  
一緒にDOMを勉強しましょう。

# 3-7: 初心者卒業！DOM操作！！

---

DOM操作、それはJSからHTMLを扱う仕組みです

これを聞いて記憶力のある方は「やったことあるゾ」となったと思います。

# 3-7: 初心者卒業！DOM操作！！

---

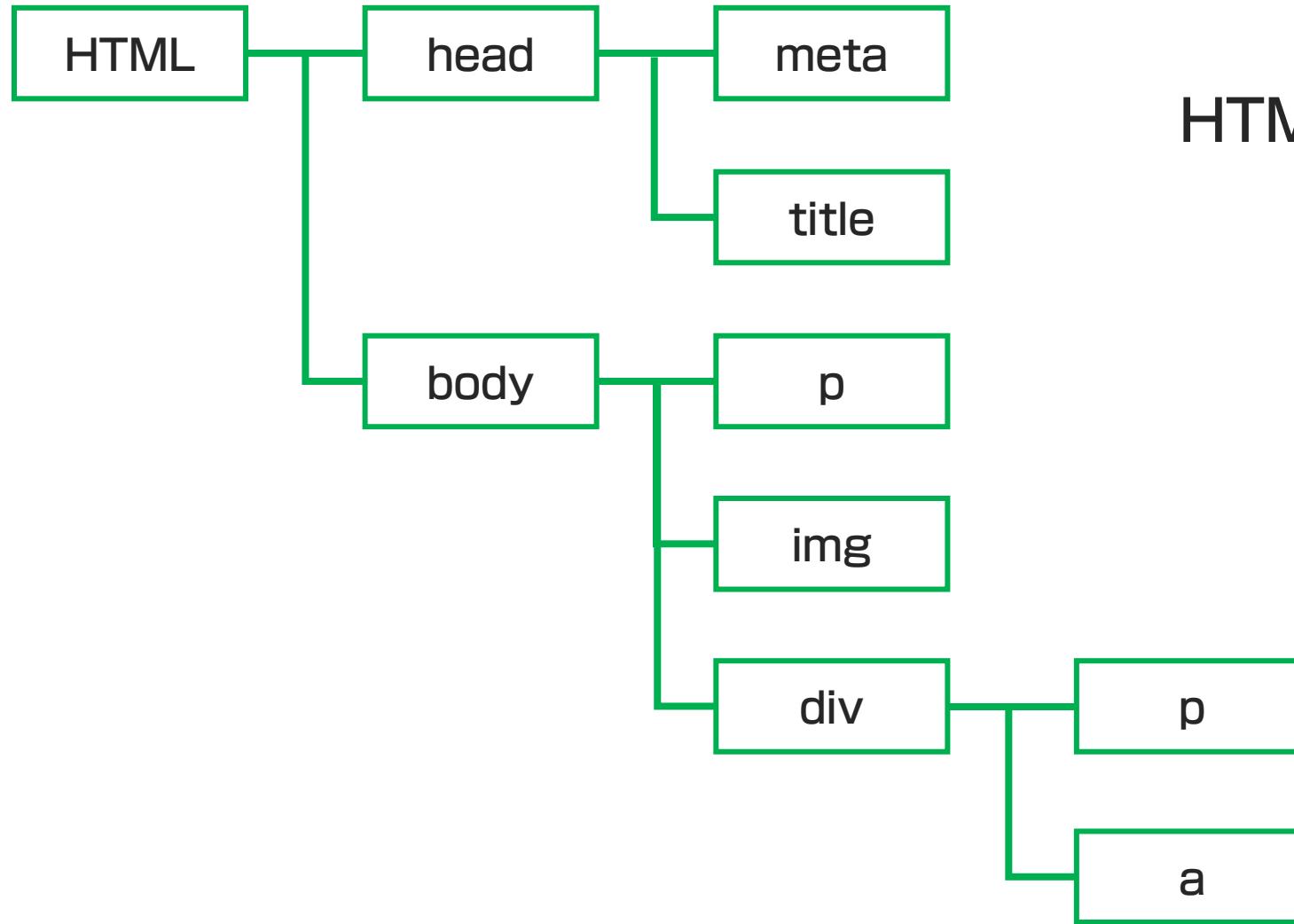
実はif文をお勉強したときに出でてきたこれ！

```
document.getElementById(" ").innerHTML = "aaaa";
```

JSからHTMLを扱っているのでDOM操作なのです！！

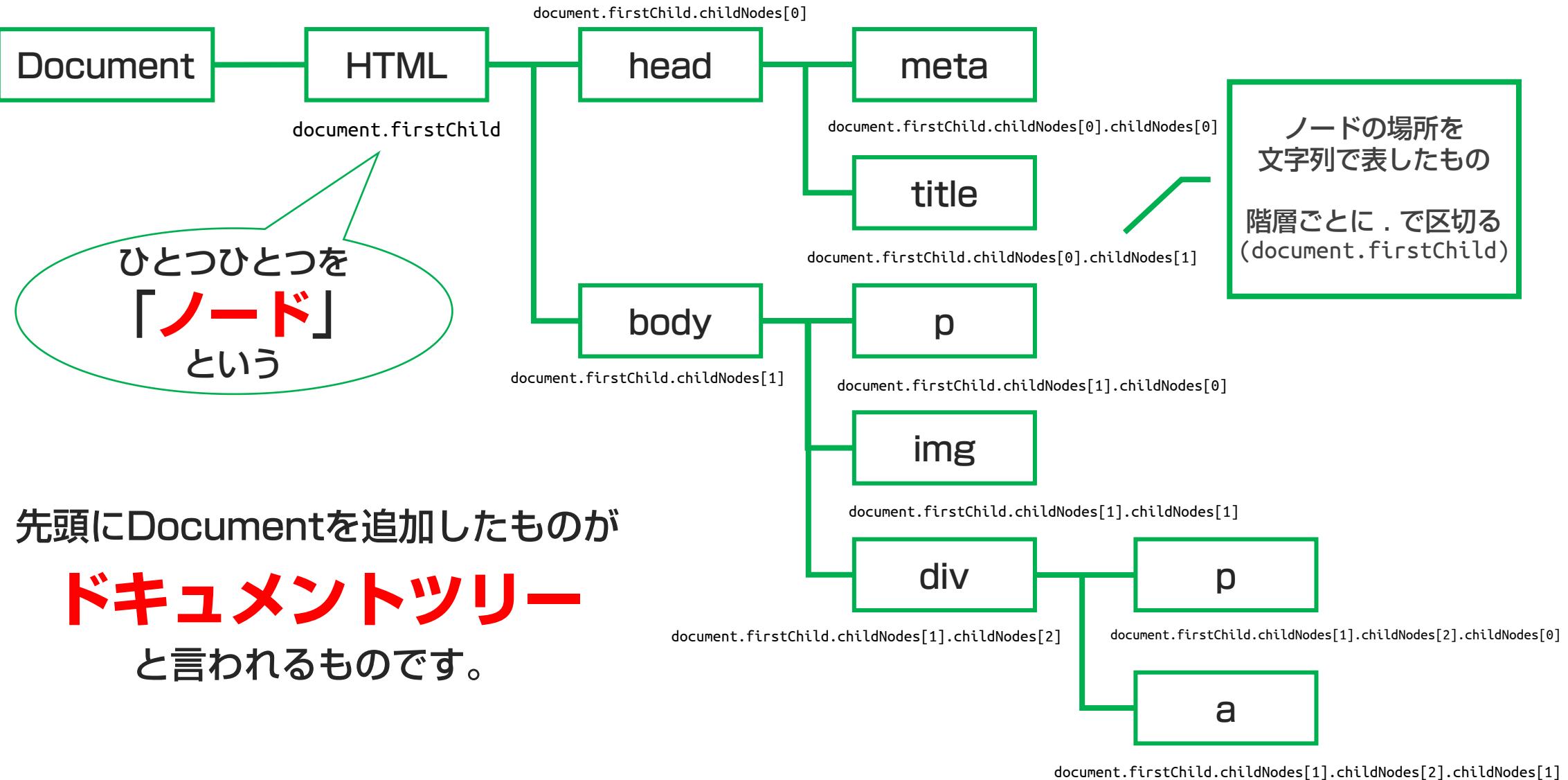
このようにJSからHTMLを操作するのをDOM操作といいます。  
FizzBuzz問題でもDOM操作やってたね。

## 3-7-2: ドキュメントツリー



先程説明したとおり  
HTMLは階層構造になっています  
(右図のように表せる)

## 3-7-2: ドキュメントツリー



# 3-7: 初心者卒業！DOM操作！！

---

DOM操作で扱うノードの種類は主に3つです。

- **要素ノード**  
→HTMLタグのこと
- **属性ノード**  
→srcとか idとか hrefとかの HTMLプロパティのこと
- **テキストノード**  
→上記以外のテキスト

# 3-7: 初心者卒業！DOM操作！！

```
1 <!DOCTYPE html>
2 <html>
3
4 <head>
5   <meta charset="utf-8">
6   <title>たいとる</title>
7 </head>
8
9 <body>
10  <div id="block1">ブロック1</div>
11  <p>テキスト</p>
12 </body>
13
14 </html>
```

では、左のHTMLの  
divの中身をDOMで表そう

childNodesよりchildrenを使ったほうがいいかも

Document.firstChild.childNodes[1].childNodes[0].nodeValue

<html>

<body>

<div>

ブロック1

■要素ノード

■属性ノード

■テキストノード

# 3-7: 初心者卒業！DOM操作！！

---

あれ？意外と簡単じゃね？？

そうなんです！実は難しくないんです！！

# 3-7: 初心者卒業！DOM操作！！

---

でも、これって結構面倒くさくないですか……？

# 3-7: 初心者卒業！DOM操作！！

---

実は、今まで黙ってたんですが  
いい感じに書ける方法があるんです。  
これからそれを伝授してしんぜよう。

# 3-7: 初心者卒業！DOM操作！！

```
dom.html x calc.html
04回目 › dom.html › html
1 <!DOCTYPE html>
2 <html>
3
4 <head>
5   <meta charset="utf-8">
6   <title>たいとる</title>
7   <link rel="stylesheet" href="./CSS/style.css">
8 </head>
9
10 <body>
11   <div>ブロック1</div>
12   <div>ブロック2</div>
13   <div>ブロック3</div>
14 </body>
15
16 </html>
```

左の内容で  
dom.htmlを作ろう！

# 3-7: 初心者卒業！DOM操作！！

```
5 dom.html ✘ 5 calc.html
04回目 ➔ 5 dom.html ➔ html
1  <!DOCTYPE html>
2  <html>
3
4  <head>
5      <meta charset="utf-8">
6      <title>たいとる</title>
7      <link rel="stylesheet" href="./CSS/style.css">
8  </head>
9
10 <body>
11     <div>ブロック1</div>
12     <div>ブロック2</div>
13     <div>ブロック3</div>
14 </body>
15
16 </html>
```

2つ目のdivを参照する！  
(タグ名で参照する)

document.getElementsByTagName("div")[1].childNodes[0].nodeValue

これで参照できる！

0スタートなので注意

("div")[0] : 1つめ  
("div")[1] : 2つめ  
("div")[2] : 3つめ  
.....

配列と同じ！！

# 3-7: 初心者卒業！DOM操作！！

2つ目のdivの内容を出力してみよう！  
(警告ウインドウで出力しよう！)

```
document.getElementsByTagName("div")[1].childNodes[0].nodeValue
```

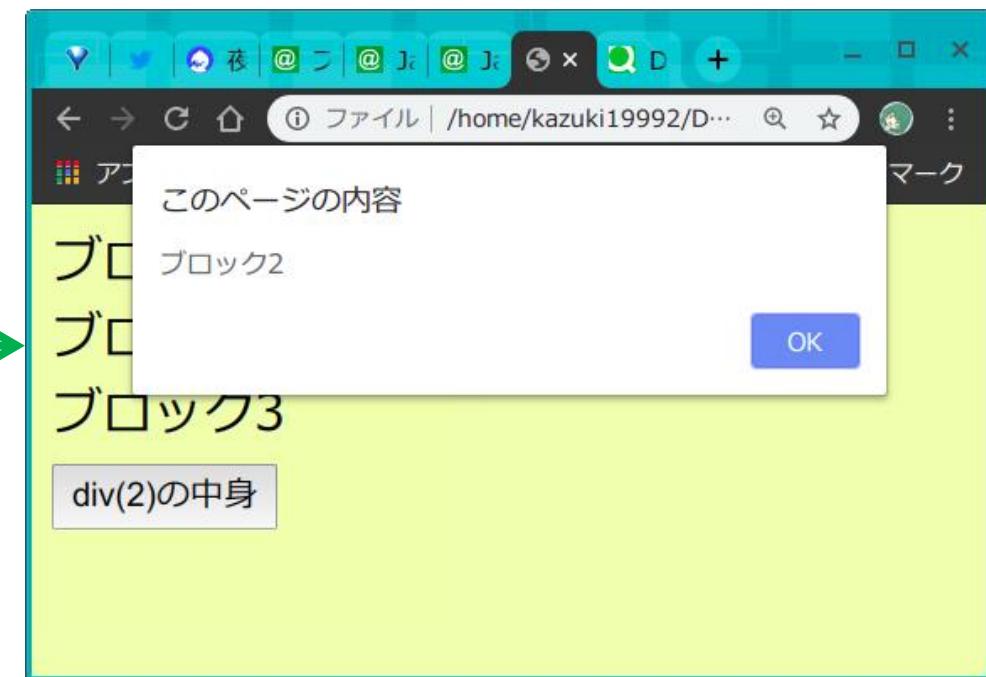
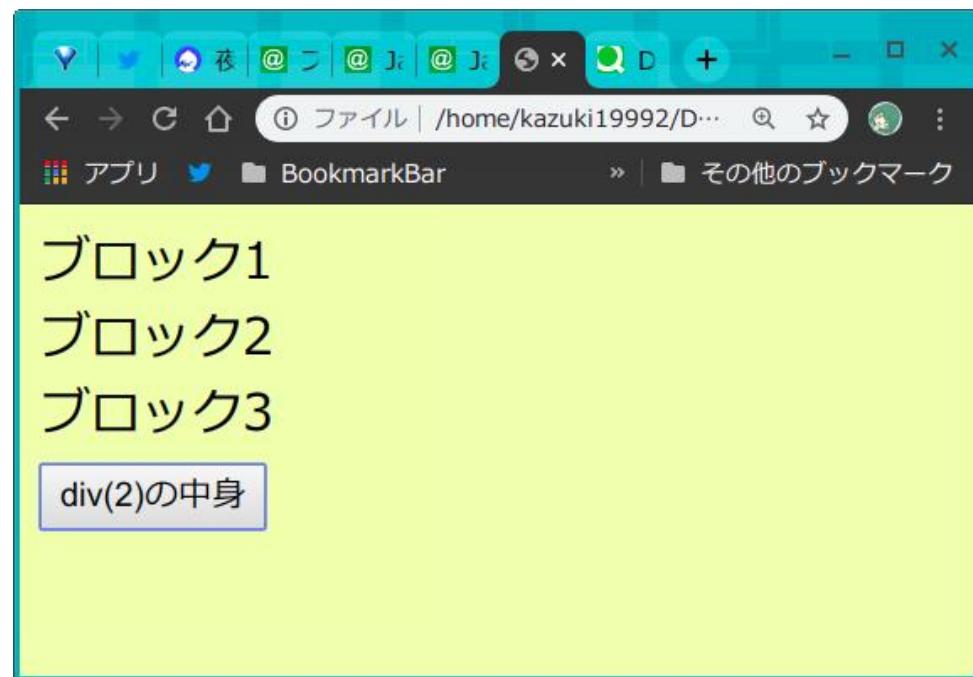
The screenshot shows a code editor interface with the following details:

- Title Bar:** Shows two tabs: "dom.html" and "event.html".
- Breadcrumb:** Shows the path: "04回目" → "dom.html" → "html".
- Code Area:** Displays the following code in a syntax-highlighted editor:

```
9
10 <body>
11   <div>ブロック1</div>
12   <div>ブロック2</div>
13   <div>ブロック3</div>
14
15   <!-- クリックすると2つ目のdivの中身が表示される -->
16   <input type="button" value="div(2)の中身" onclick='alert(document.getElementsByTagName("div")[1].childNodes[0].nodeValue)'>
17 </body>
```

# 3-7: 初心者卒業！DOM操作！！

ブラウザで実行してみよう！！  
上手く行けば成功だ！！



# 3-7: 初心者卒業！DOM操作！！

---

もっと楽な方法があります。  
参照するタグにIDをつけてあげましょう。  
(今回はブロック3にIDをつける)

```
10 <body>
11   <div>ブロック1</div>
12   <div>ブロック2</div>
13   <div id="block3">ブロック3</div>
14
```

# 3-7: 初心者卒業！DOM操作！！

---

IDで参照しよう！！(出力は警告ウィンドウ)

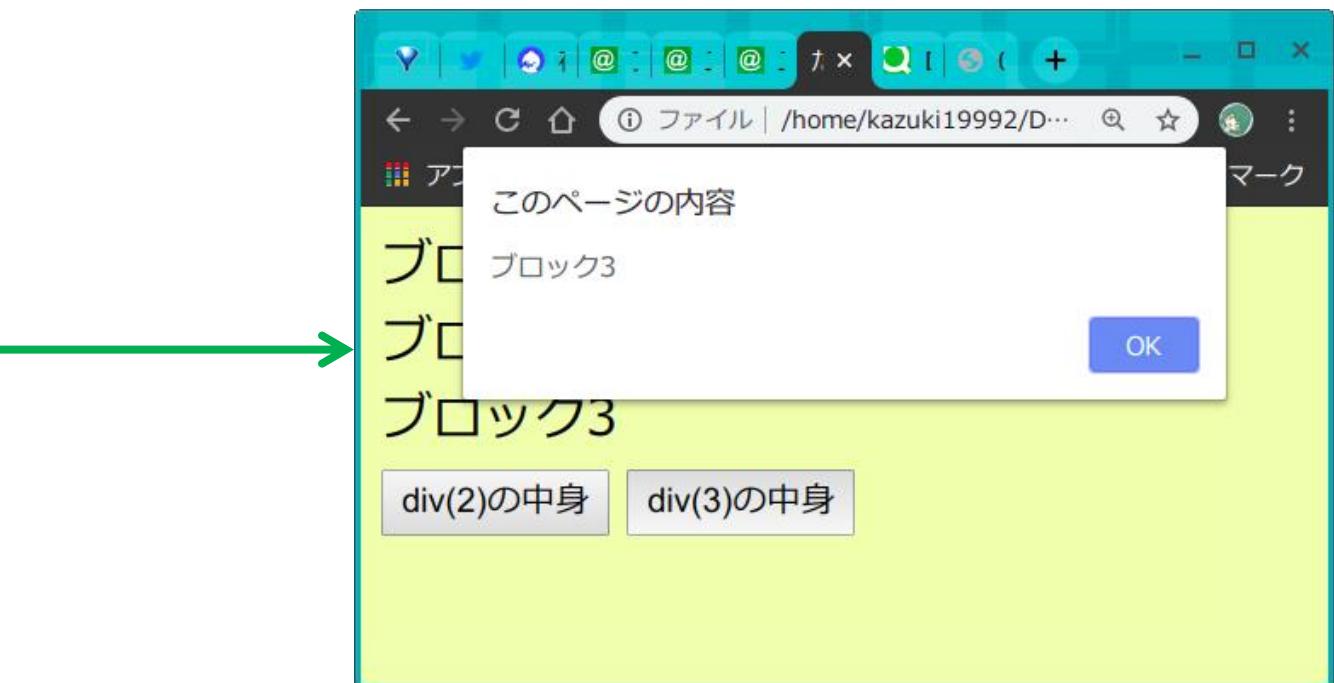
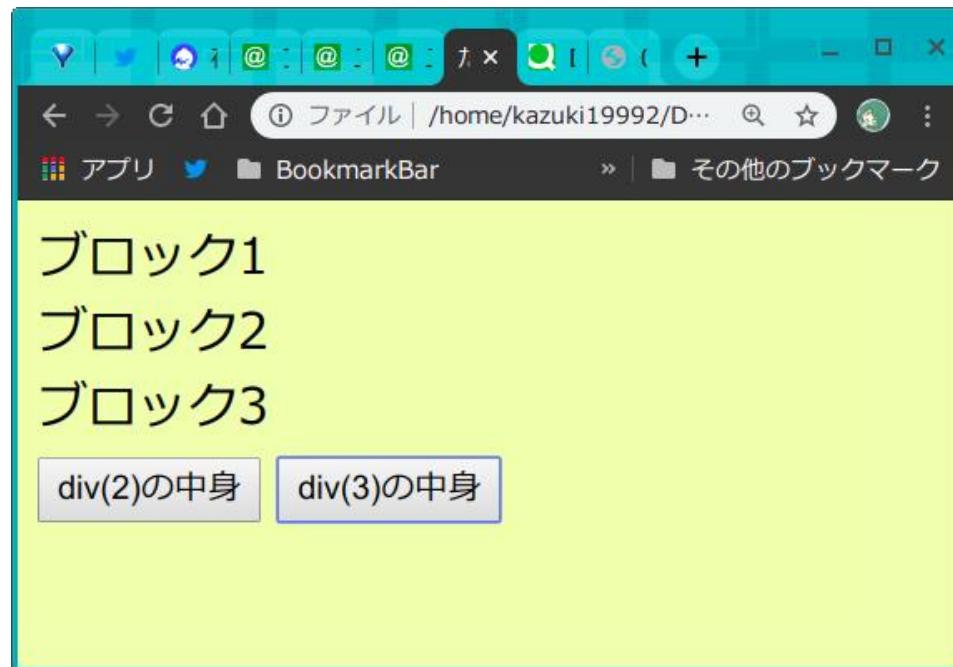
document.getElementById("block3").childNodes[0].nodeValue

getElementById("id名")に参照するIDを書こう！

```
13 <div id="block3">ブロック3</div>
14
15 <!-- クリックすると2つ目のdivの中身が表示される -->
16 <input type="button" value="div(2)の中身" onclick='alert(document.getElementsByTagName("div")[1].childNodes[0].nodeValue)'>
17
18 <!-- クリックすると3つ目のdivの中身が表示される -->
19 <input type="button" value="div(3)の中身" onclick='alert(document.getElementById("block3").childNodes[0].nodeValue)'>
20
21
22 </body>
```

# 3-7: 初心者卒業！DOM操作！！

ブラウザで `div(3)の中身` をクリックしてしっかりと  
「ブロック3」の文字を押せたら成功だ！！



# 3-7: 初心者卒業！DOM操作！！

---

ここまでついてこれた君達は優秀だ！！  
お疲れ様！！

# 3-7: 初心者卒業！DOM操作！！

---

……と、言いたいところだが

要素の取得までできたのに、それで終わるのはもったいなくない？  
せっかくなら書き換えまでやってみよう！！

# 3-7: 初心者卒業！DOM操作！！

<script>タグと入力フォームを追記しよう！

```
6   <title>たいとる</title>
7   <link rel="stylesheet" href="./CSS/style.css">
8 </head>
9 <script type="text/javascript" src="JS/script_7.js"></script>
10
11 <body>
12   <!-- ボタンを追加 -->
13
14
15
16
17
18
19
20
21
22   <!-- 書き換え -->
23   <div class="rewriting">
24     <!-- 何番目のdivを書き換えるか(1から3まで) -->
25     何個目？(1から3まで) <input type="text" id="num">
26     <BR>
27
28     <!-- どんなデータに書き換えるか -->
29     データは?<input type="text" id="data">
30     <BR>
31
32     <!-- クリックすると書き換えが実行される -->
33     <input type="button" value="書き換える！" onclick='rewriting(num.value, data.value)'>
34   </div>
35
36 </body>
```

# 3-7: 初心者卒業！DOM操作！！

script\_7.jsに関数rewriting(num, data)を作ろう！

num番目のdivをdataに書き換えているよ！！



```
dom.html      JS script_7.js ×
04回目 › JS › JS script_7.js › rewriting
1 function rewriting(num, data) {
2     // numは1から3なので0から2に修正する
3     num--;
4
5     // 書き換え処理
6     document.getElementsByTagName("div")[num].childNodes[0].nodeValue = data;
7
8     // 完了通知
9     alert("書き換えに成功しました！！");
10 }
```

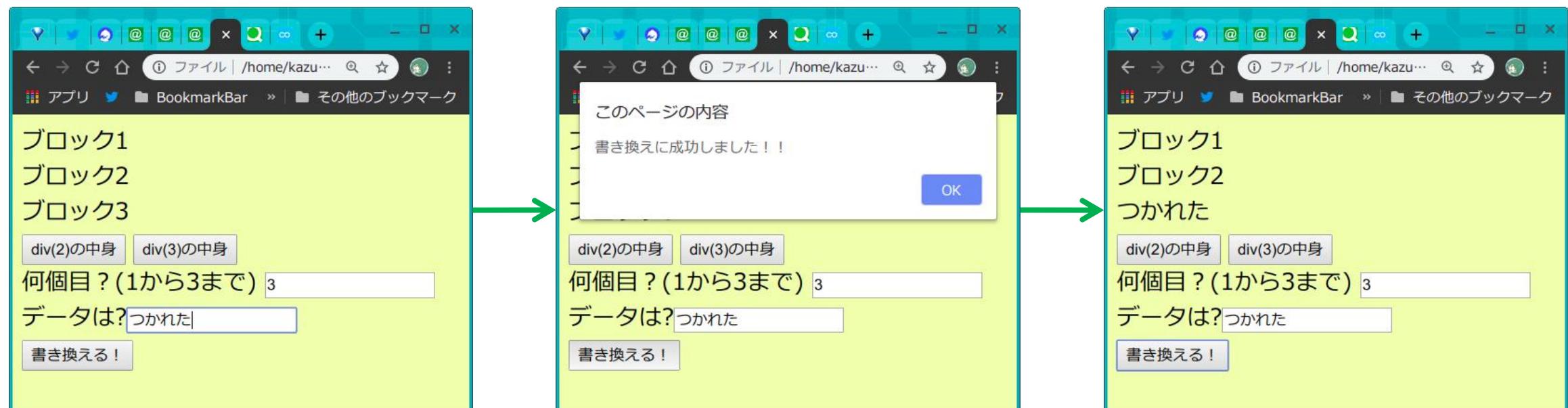
The screenshot shows a code editor with two tabs: "dom.html" and "script\_7.js". The "script\_7.js" tab is active, displaying the following code:

```
function rewriting(num, data) {
    // numは1から3なので0から2に修正する
    num--;
    // 書き換え処理
    document.getElementsByTagName("div")[num].childNodes[0].nodeValue = data;
    // 完了通知
    alert("書き換えに成功しました！！");
}
```

The code uses ES6 syntax (arrow functions) and includes comments explaining the logic: it takes a number `num` and a string `data`, finds the `div` element at index `num` (adjusted from 1 to 0), and changes its child node's value to `data`. It also includes an alert message confirming success.

# 3-7: 初心者卒業！DOM操作！！

ブラウザで実際に動かしてみよう！！！  
こんな感じに上手く動けば大成功！おめでとう！！



# 4: おわり

---

スライド番号ちょうど100です……

つかれた(小声)

ここまで頑張った皆さん、本当にお疲れ様でした。

あなたはもう立派なJSテクニシャンです(は?)

これをマスターすればあとはネットからコピペしたJSコードでも  
難無く修正して動かすことができると思います

合宿当日がんばりましょー！！