

プログラミング

第4回

JavaScript に触れてみよう

久保田 匠

[準備]授業資料にアクセス

いつもの作業

- 久保田の授業ホームページに資料がアップロードされている。
- まずは「愛教大 数学」と検索してみよう。



A screenshot of the Kubota teaching homepage. At the top, it says '授業用ホームページ (久保田)' and '2025年度前期担当科目'. Below is a table showing the weekly schedule for the first semester:

	月曜	火曜	水曜	木曜	金曜
1限					
2限	確率統計Ⅱ			確率統計Ⅱ	
3限				線形数学演習Ⅰ	確率統計Ⅱ
4限	4年ゼミ				(オフィスアワー)
5限					

Below this is another table for the second semester:

	月曜	火曜	水曜	木曜	金曜
1限					
2限					
3限	科学リテラシー				
4限	(オフィスアワー)	3年ゼミ		4年ゼミ	プログラミング プログラミング
5限					

At the bottom, it says '他のコンテンツ → ● ●' and provides contact information: '数学教育講座 久保田匠 (自然科学棟 521 研究室)
Email: skubota [at] auecc.aichi-edu.ac.jp'. To the right of the homepage screenshot, there is a large grey arrow pointing left.

	内容	資料	コード
第1回	いろいろなプログラミング言語 VSCode のインストール	● Prog_01-1	
第2回	Webページを構築する(HTML)	● Prog_02-1	
第3回	Webページの見栄えを整える(CSS)	Prog_03-1 Prog_03-2	
第4回	JavaScriptに触れてみよう	● Prog_04-1	
第5回	変数と演算		(なし)
第6回	条件文		(なし)
第7回	繰り返し(1)		(なし)
第8回	繰り返し(2)		Prog_08-1
第9回	繰り返し(3)		(なし)

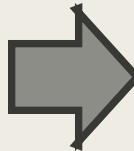
[準備] コードの新規作成①

いつもの作業

- 授業用ホームページからサンプルコードをコピーしよう。

プログラミング

	内容	資料	コード
第1回	いろいろなプログラミング言語 VSCode のインストール	●	Prog_01-1
第2回	Webページを構築する(HTML)	●	Prog_02-1
第3回	Webページの見栄えを整える(CSS)		Prog_03-1 Prog_03-2
第4回	JavaScriptに触れてみよう	●	Prog_04-1
第5回	変数と演算		(なし)
第6回	条件文		(なし)
第7回	繰り返し(1)		(なし)
第8回	繰り返し(2)		Prog_08-1
第9回	繰り返し(3)		(なし)



Prog_04-1

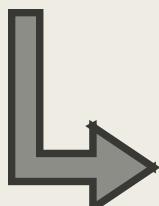
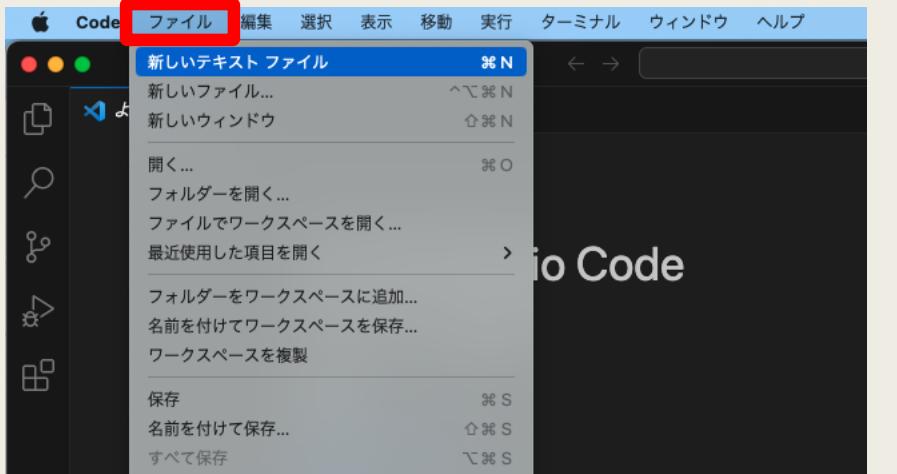
```
<!DOCTYPE html>
<html>
<head>
    <meta charset="UTF-8">
    <title>Prog_04-1</title>
    <!-- しばらくここは使いません。 -->
</head>
<body>
    <!-- ここに今日の授業内容を入力します。 -->
</body>
</html>
```

コピー

[準備] コードの新規作成②

いつもの作業

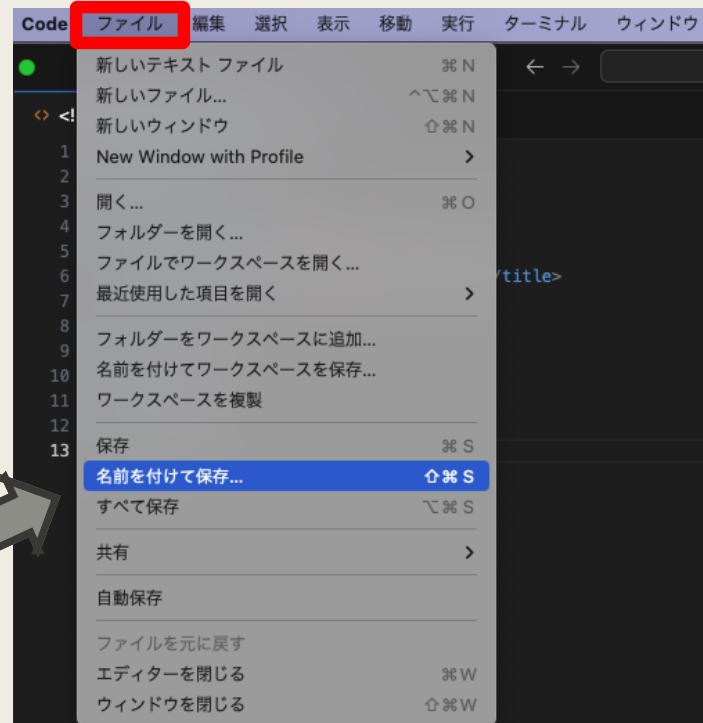
- VSCode を起動し「ファイル」から「新しいテキストファイル」を選択。
- その後、さきほどコピーした文書をペースト（Ctrl + V）して「名前をつけて保存」。



Ctrl + V

A screenshot of the VSCode editor showing an HTML file named 'Untitled-1'. The code in the editor is:

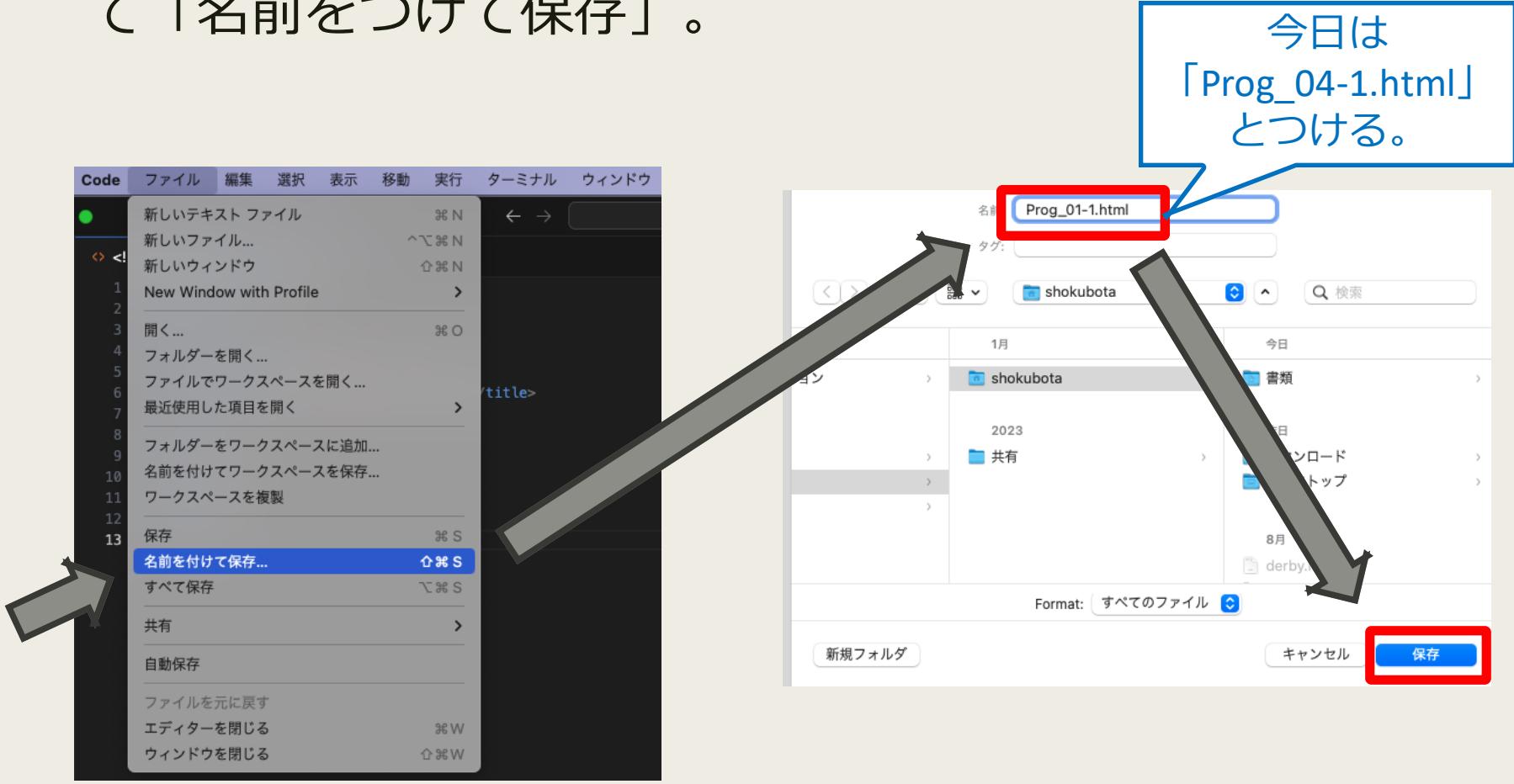
```
<!DOCTYPE html>
<html>
<head>
    <meta charset="UTF-8">
    <title>ここにページのタイトルを書きます</title>
</head>
<body>
    こんにちは！
</body>
</html>
```



[準備] コードの新規作成②

いつもの作業

- VSCode を起動し「ファイル」から「新しいテキストファイル」を選択。
- その後、さきほどコピーした文書をペースト (Ctrl + V) して「名前をつけて保存」。



[準備]作業環境を整える

いつもの作業

- 保存したhtmlファイルをダブルクリックして開いておく。
- PCの画面をふたつに分け、片方はブラウザ、もう片方はvsCodeを開いておくと便利。



A screenshot of a Mac desktop showing two windows side-by-side. The left window is a web browser displaying the same HTML content as the previous screenshot. The right window is a code editor (VS Code) showing the same HTML code. The code editor has a dark theme and includes status bar information at the bottom: '行 13, 列 8 スペース: 4 UTF-8 CRLF HTML'.



[復習]HTML

- JavaScript は Web ページで動くプログラミング言語。
- JavaScript を使うためにはまず Web ページを構築する必要がある。
- Web ページを構築するために HTML を用いる。
- HTML は Web ページの土台を作るためのツール。



The screenshot shows a code editor window with the title "Prog_01-1.html". The file path is "Users > shokubota > Documents > 授業支援サイト > Prog_01-1.html". The code itself is:

```
<!DOCTYPE html>
<html>
<head>
    <meta charset="UTF-8">
    <title>ここにページのタイトルを書きます</title>
</head>
<body>
    こんにちは！<br>
    私の名前は久保田匠です。
</body>
</html>
```

[復習]Webページの見栄えを整える

- HTML は Web ページの構造を記述する。
- しかし、見栄えの点では、物足りない部分や使い勝手の悪い点が少なくない。
- CSS は Web ページの見た目（色や配置、余白など）を自由に整えるための仕組み。
 - CSS は Cascading Style Sheets の略。

全部中央揃えにしたい

	大人	子ども
入場料	2000円	1000円
限定アトラクション	500円	300円

めんどくさい('A')

	大人	子ども
入場料	2000円	1000円
限定アトラクション	500円	300円

```
9 <body>
10   <table border="1">
11     <tr>
12       <td>□</td>
13       <td><center>大人</center></td>
14       <td><center>子ども</center></td>
15     </tr>
16     <tr>
17       <td><center>入場料</center></td>
18       <td><center>2000円</center></td>
19       <td><center>1000円</center></td>
20     </tr>
21     <tr>
22       <td><center>限定アトラクション</center></td>
23       <td><center>500円</center></td>
24       <td><center>300円</center></td>
25     </tr>
26   </body>
```

```
1  <!DOCTYPE html>
2  <html>
3
4  <head>
5    <meta charset="UTF-8">
6    <title>Prog_03-0</title>
7    <style>
8      td {
9        text-align: center;
10      }
11    </style>
12  </head>
13  <body>
14    <table border="1">
15      <tr>
16        <td>□</td>
17        <td>大人</td>
18        <td>子ども</td>
19      </tr>
20      <tr>
21        <td>入場料</td>
22        <td>2000円</td>
23        <td>1000円</td>
24      </tr>
25      <tr>
26        <td>限定アトラクション</td>
27        <td>500円</td>
28        <td>300円</td>
29      </tr>
30    </table>
31  </body>
32 </html>
```

[復習] style 属性を利用してcssを使う

- CSSの最も簡単な使い方は HTML のタグ内で **style 属性** を使う方法。
- このやり方は HTML のみで見栄えを整えるやり方とさほど変わらないが、基本的にはこれができるれば css については問題なし。

style 属性

```
<!DOCTYPE html>
<html>
<head>
    <meta charset="UTF-8">
    <title>Prog_03-2</title>
</head>

<body style="text-align: center">
    <p style="color: red">赤色のテキスト</p>
    <p style="color: green; font-size: 24px;">緑色、24px</p>
    <p>
        テキストの一部にスタイルを適用する場合は、
        <span style="text-decoration: underline">span要素</span>を使う。
    </p>
</body>
</html>
```

- <タグ名 **style= “ ** ”**> で使うのが基本。
- <body>タグで使うと 文章全体に適用される。
- 11行目のように複数 指定したい場合はセミコロン（;）をつける。

[復習]プロパティ

- <タグ名 style=" ** "> の ** の部分をプロパティ という。
- 主なプロパティを列挙するが覚える必要はなく「たしかこんな指定ができたなあ」とうっすら把握しておけばよい。
- 大事なことは「調べれば使える」という状態にしておくこと。

プロパティ名	用途	使用例
background-color	背景色の指定	background-color: blue;
font-size	字の大きさを指定。px, ptなどの単位がある。smallなども可	font-size: 12px; font-size: small;
opacity	半透明の度合いを0~1の範囲で指定	opacity: 0.6;
text-align	left(左寄せ)、right(右寄せ)など、文字の位置を指定	text-align: center;
text-decoration	underline(下線)、underline dotted(点線下線)など、テキストの装飾敵な線を表示	text-decoration: underline;

cssはここまで理解
できていればひとまずOK

中身に入る前に... (インデントの話)

- これから、より複雑なコードを書く場面が増える。
- コードを見やすくするために インデント を意識しよう。
 - インデントとは、空白のことで、プログラムをかくときは通常 Tabキー1つ分、または半角スペース2つ分か4つ分が使われる。
- インデントの幅は自由に決めてよいが統一させること。
- VSCode では改行に応じてインデントが自動で整えられるが、場合によっては手動で調整が必要になることもある。

```
<body>
<p>
<script>
var point = prompt("試験の点数を0から100以下の整数値で入力してください。");
document.write("あなたの成績は" + point + "点でした。")
if(point >= 60){
document.write("単位取得おめでとう！")
} else {
document.write("不合格です。また来年頑張ってください。")
}
</script>
</p>
</body>
```

インデントがないコード
(構造が分かりにくい)

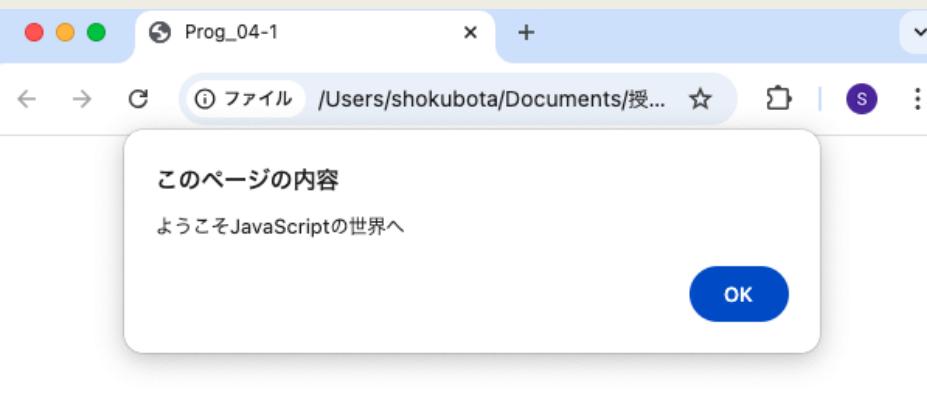
```
<body>
<p>
<script>
    var point = prompt("試験の点数を0から100以下の整数値で入力してください。");
    document.write("あなたの成績は" + point + "点でした。")
    if(point >= 60){
        document.write("単位取得おめでとう！")
    } else {
        document.write("不合格です。また来年頑張ってください。")
    }
</script>
</p>
</body>
```

インデントがあるコード
(構造が分かりやすい)

JavaScript に触れてみよう

- JavaScript の役割は Web ページに「動き」を与えること。
 - ボタンをクリックしたときに何かが起こるようにする。
 - フォームに入力されたデータをチェックする。
 - 時間ごとに表示が変わるアニメーションを作る。
- JavaScript の命令は <script> タグ内に記述する。
- 次のコードを入力してみよう。

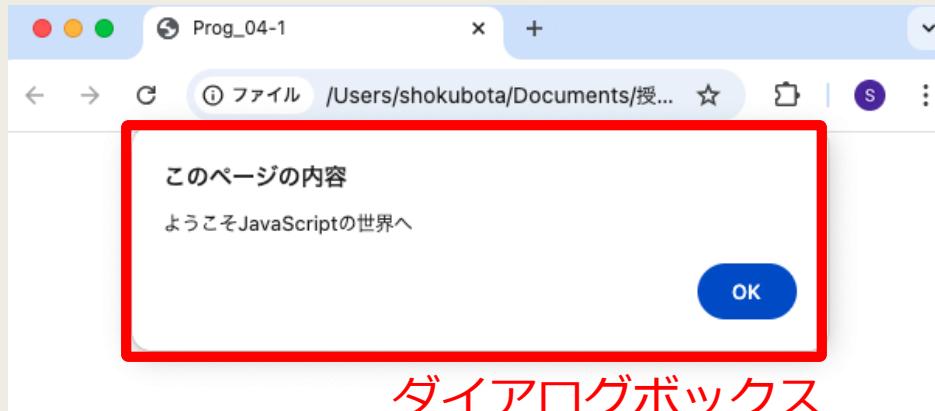
```
1  <!DOCTYPE html>
2  <html>
3
4  <head>
5      <meta charset="UTF-8">
6      <title>Prog_04-1</title>
7  </head>
8
9  <body>
10     <script>
11         alert("ようこそJavaScriptの世界へ");
12     </script>
13  </body>
14
15  </html>
```



ステートメント

- プログラムではひとつの命令の単位をステートメントという。
- 先ほど使った alert はダイアログボックスに何かを表示させる命令。
- 文字列を扱う場合はタブルクオート(“”)と(“”)で囲む。
- ステートメントの終わりにはセミコロン（;）をつける。
 - JavaScript ではコード内で適切に改行されていればセミコロンを省いても動くが、初学者は行儀よくしておこう。

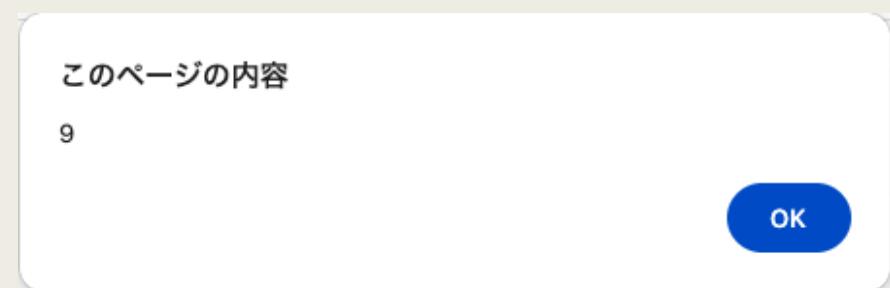
```
1  <!DOCTYPE html>
2  <html>
3
4  <head>
5      <meta charset="UTF-8">
6      <title>Prog_04-1</title>
7  </head>
8
9  <body>
10     <script>
11         alert("ようこそJavaScriptの世界へ");
12     </script>
13  </body>
14
15  </html>
```



簡単な計算をしてみよう

- 先のコードに次を追加してみよう。

```
1  <!DOCTYPE html>
2  <html>
3
4  <head>
5      <meta charset="UTF-8">
6      <title>Prog_04-1</title>
7  </head>
8
9  <body>
10     <script>
11         alert("ようこそJavaScriptの世界へ");
12         alert(4+5);
13     </script>
14 </body>
15
16 </html>
```



- 「4+5」が計算され、計算結果の「9」がダイアログボックスに表示された。

引数

- 命令に与える何らかの「値」のことを引数(ひきすう)という。

```
alert("ようこそJavaScriptの世界へ");  
alert(4+5);
```

- 上の例で、1行目は引数がダブルクオートで囲まれているが、2行目はダブルクオートで囲まれていない。
- JavaScript では1行目の引数は「文字列」と判断し、2行目の引数は「数値」と判断する。
- したがって、2行目では「4+5」そのものではなく、その計算結果である「9」が出力された。
- 「4+5」そのものを出力させたい場合は alert("4+5"); と入力する。

[余談]コメントアウト

- HTMLにおいて、`<!--` と `-->` で囲まれた部分をコンピュータは処理しないのだった（第2回の資料p17）。
- 同様の機能が JavaScript にもあり、`<script>`タグ内のコメントアウトは次のように行う。

```
1  <!DOCTYPE html>
2  <html>
3
4  <head>
5      <meta charset="UTF-8">
6      <title>Prog_04-1</title>
7  </head>
8
9  <body>
10     <!-- <script>タグで囲まれていない箇所はこのように HTML のコメント機能を使う。 -->
11     <script>
12         /* <script>タグで囲まれている箇所はこのようにするとコメントをつけられる。 */
13         // alert("ようこそ JavaScript の世界へ");
14         alert(4+5);
15     </script>
16  </body>
17
18  </html>
```

緑字で表示されている箇所は実行されない。

- 12行目のように `/*` と `*/` 囲まれた部分は処理されない。
- 13行目のように `//` を使うと箇所以降行末までは処理されない。

文字列の足し算

- $1 + 1 = 2$ である。
- では、文字列の足し算 “1” + “1” の結果はどうなるだろうか？
- 次のコードを入力してみよう。

```
1  <!DOCTYPE html>
2  <html>
3
4  <head>
5  |   <meta charset="UTF-8">
6  |   <title>Prog_04-1</title>
7  </head>
8
9  <body>
10 |  <script>
11 |  |  alert("1" + "1");
12 |  </script>
13 </body>
14
15 </html>
```

- 正解は「11」。
- 文字列の足し算はふたつの文字列を連結させる。

文字列 + 数値

- では、文字列と数値を足し算してみるとどうなるだろうか？
- 次のコードを入力してみよう。

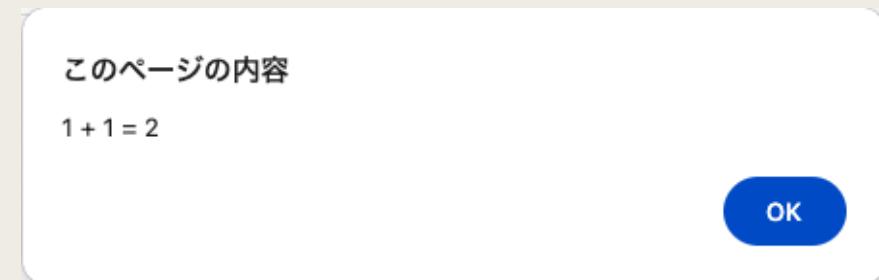
```
1  <!DOCTYPE html>
2  <html>
3
4  <head>
5  |   <meta charset="UTF-8">
6  |   <title>Prog_04-1</title>
7  </head>
8
9  <body>
10 |  <script>
11 |  |  alert("1" + 1);
12 |  </script>
13 </body>
14
15 </html>
```

- 正解は「11」。
- 文字列 + 数値 を計算させると JavaScript の場合は、数値を自動的に文字列に変換して計算（連結）する。
- プログラミング言語によってはエラーになる。

“1 + 1 = ” + (1+1)

■ 次のコードを入力してみよう。

```
1  <!DOCTYPE html>
2  <html>
3
4  <head>
5      <meta charset="UTF-8">
6      <title>Prog_04-1</title>
7  </head>
8
9  <body>
10 <script>
11     alert("1 + 1 = " + (1+1));
12 </script>
13 </body>
14
15 </html>
```



■ コンピュータがどのように処理しているか考えてみよう。

カッコの中は数値同士の足し算なので普通に計算

alert("1 + 1 = " + (1+1));

alert("1 + 1 = " + 2);

文字列 + 数値 なので
2 を文字列に変換

alert("1 + 1 = " + “2”);

文字列同士の
足し算なので連結

ブラウザに出力させる

- ダイアログボックスではなくブラウザ（画面）に出力させるには `document.write(***)` という命令を使う。
- 次のコードを入力してみよう。

```
1  <!DOCTYPE html>
2  <html>
3
4  <head>
5      <meta charset="UTF-8">
6      <title>Prog_04-1</title>
7  </head>
8
9  <body>
10     <h1>ようこそJavaScriptの世界へ</h1>
11     <p>
12         計算結果は
13         <script>
14             document.write(12 + 7);
15         </script>
16         です。
17         </p>
18     </body>
19
20 </html>
```

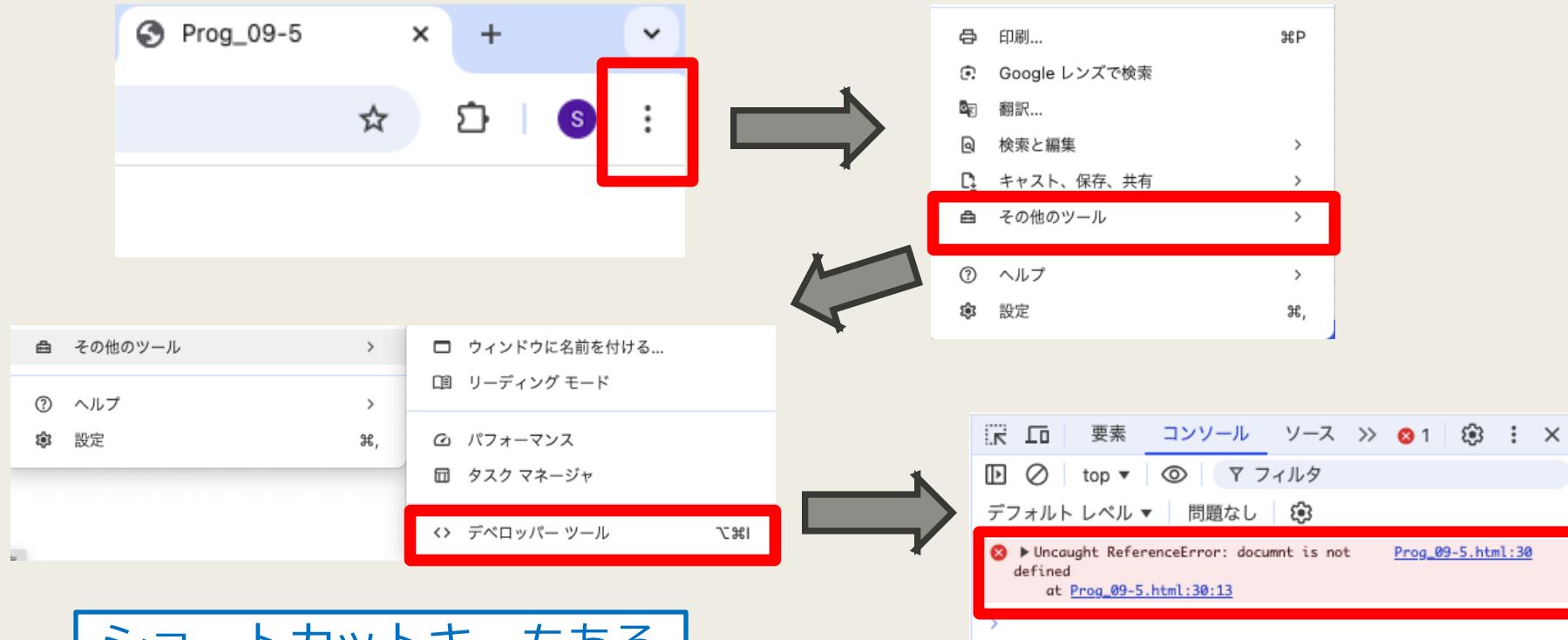
ようこそJavaScriptの世界へ

計算結果は 19 です。

デベロッパーツール

ブラウザによって呼び方が違う。
「開発者ツール」「開発メニュー」など

- 画面に何も表示されないときや、途中までしか表示されないときはプログラムに間違いがある可能性が高い。
- そのときは「デベロッパーツール」を開き、何行目でエラーが発生しているか見てみよう。



ショートカットキーもある

Windows → Ctrl + Shift + i

Mac → Option + Command + i

30行目でエラーが発生。
document が未定義と言われている
(スペルミスが発生していた)

デベロッパーツールに出力させる

- デベロッパーツールに出力させることもできる。
 - 開発中のプログラムが正しく動いているかを途中で確認したり、エラーの原因を調べるときに使う。
- その場合は console.log(***) という命令を使う。
- 次のコードを入力してみよう。

```
1  <!DOCTYPE html>
2  <html>
3
4  <head>
5      <meta charset="UTF-8">
6      <title>Prog_04-1</title>
7  </head>
8
9  <body>
10     <h1>ようこそJavaSctiptの世界へ</h1>
11     計算結果は
12     <script>
13         //document.write(12 + 7);
14         console.log(12 + 7);
15     </script>
16     です。
17 </body>
18
19 </html>
```

ようこそJavaSctiptの世界へ

計算結果は です。

↑ブラウザには表示されない

↓コンソールに表示される



まとめ

ステートメント（ひとつの命令）

```
alert("ようこそJavaScriptの世界へ");
```

引数（ひきすう）

セミコロンを
忘れないで

↓JavaScriptにおけるコメントアウト

- 文字列同士の足し算 → 連結

↓JavaScriptにおけるコメントアウト

```
alert("やき" + "にく"); //やきにく
```

- 数値同士の足し算 → 計算した結果を出力

```
alert(33+4); //37
```

- 文字列 + 数値 → 数値を文字列に変換して連結

```
alert("33" + 4); //334
```

- document.write(***) → ブラウザに出力

```
document.write("やきにく")
```

- console.log(***) → デベロッパーツールのコンソールに表示

```
console.log("やきにく")
```

[演習]教科書を熟読しよう

- 今日の内容は教科書の p30～p60 がベースになっている。
- 残った時間で自分でも該当箇所を熟読してみよう。
- 授業で解説していないコードは自分でも入力してみてどのような出力結果になるか確かめてみよう。