

# プログラミング

第4回

JavaScript に触れてみよう

久保田 匠

# 授業資料について

- 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。



## 愛知教育大学 数学教育講座

	所属教員	時間割
教員と研究	<a href="#">愛知教育大学数学教育学会</a>	<a href="#">イプシロン</a>
	<a href="#">他の研究会</a>	
リンク	<a href="#">愛知教育大学</a>	<a href="#">MathSciNet</a>
	<a href="#">数学第2サーバー</a>	<a href="#">まなびネット</a>
	<a href="#">ICT教育基盤センター</a>	<a href="#">AUEリンク</a>

専任講師	Watanabe, Yuta 渡邊 悠太	有限射影幾何学	自然科学棟 523	2336	ywatanabe	
専任講師	Kubota, Sho 久保田 匠	代数的確率論	自然科学棟 521	2323	skubota	<a href="#">●</a>
助教	Ishikawa, Masaaki 石川 雅章	数学教育学	自然科学棟 535	2331	m-ishikawa	<a href="#">●</a>

● Eメールアドレスは後に、@auecc.aichi-edu.ac.jp を付けて下さい。  
● 電話番号は、内線番号です。外線からは、前に0566-26-を付けて下さい。

## 久保田匠の授業関連のページ

2024年度前期担当科目

	月曜	火曜	水曜	木曜	金曜
1限					
2限	<a href="#">確率統計II</a>	<a href="#">確率統計II</a>			
3限				<a href="#">線形数学演習</a>	<a href="#">確率統計II</a>
4限	(オフィスアワー)				
5限					

2024年度後期担当科目

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

# [準備]コードの新規作成までの流れ①

- コードの新規作成の手順はこの授業を通して共通なので記憶するように。
- 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	●	<a href="#">Prog_01-1</a>
第2回	Webページを構築する(HTML)	●	<a href="#">Prog_02-1</a>
第3回	Webページの見栄えを整える(CSS)	●	<a href="#">Prog_03-1</a> <a href="#">Prog_03-2</a>
第4回	JavaScriptに触れてみよう	●	<a href="#">Prog_04-1</a>
第5回	変数と演算	●	
第6回	剰余演算、条件文(1)	●	
第7回	条件文(2)、繰り返し(1)	●	
第8回	繰り返し(2)	●	
第9回	オブジェクト	●	



## Prog\_04-1

```
<!DOCTYPE html>
<html>

<head>
  <meta charset="UTF-8">
  <title>Prog_04-1</title>
  <!-- 今日はここは使いません。 -->
</head>

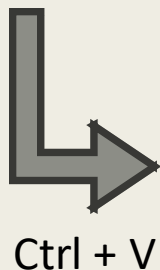
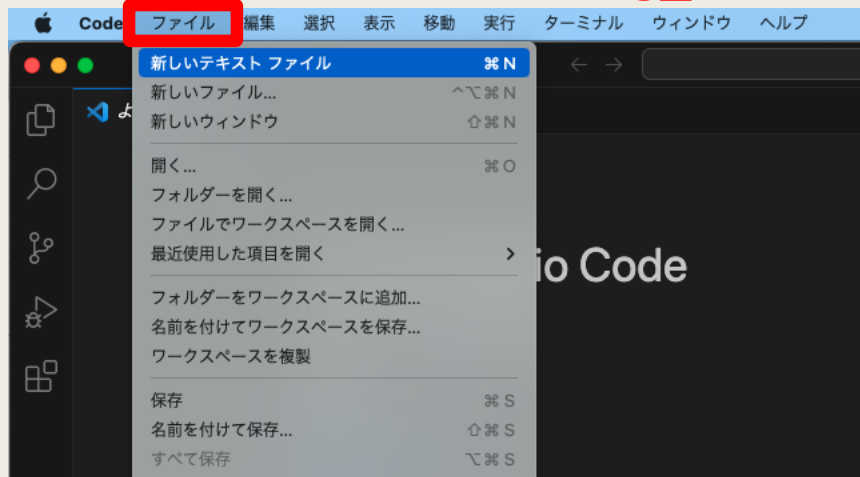
<body>
  <!-- ここに今日の授業内容を入力します。 -->
</body>

</html>
```

コピー

# [準備]コードの新規作成までの流れ②

- 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。
- そのあと、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」。
- ファイル名は「Prog\_XX-Y.html」など。



# [準備]コードの新規作成までの流れ②

- 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。
- そのあと、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」。
- ファイル名は「**Prog\_XX-Y.html**」など。

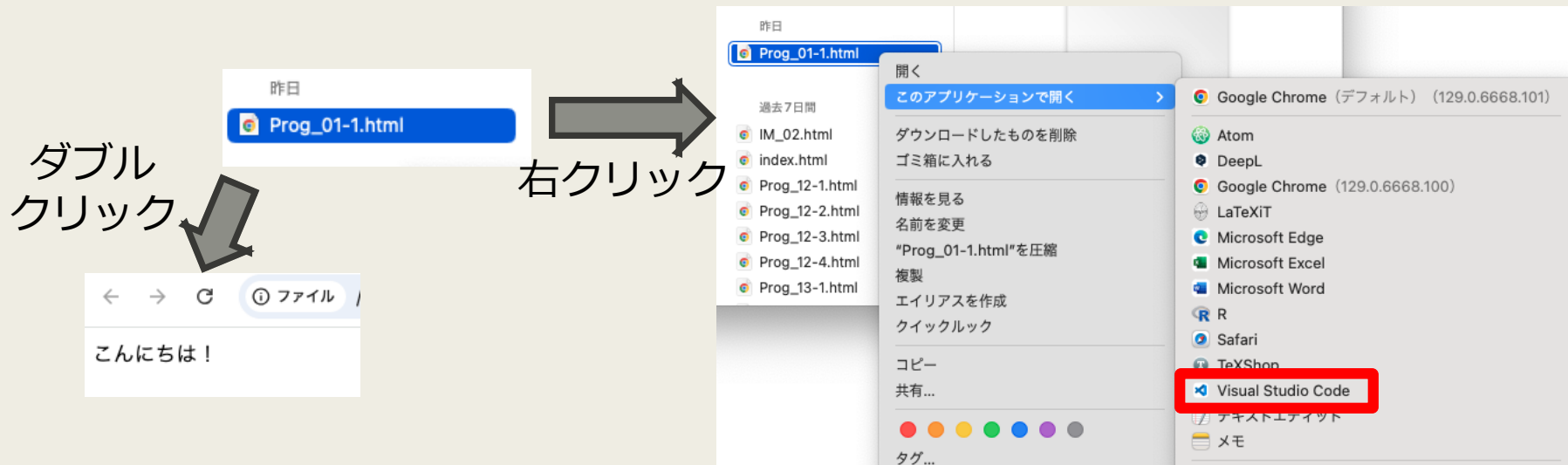
今日は  
「Prog\_04-1.html」  
とつける。



# [準備]htmlファイルを開く・編集する

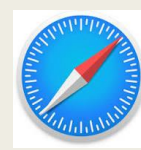
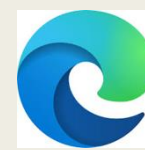
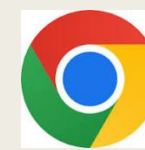
## ■ 保存したhtmlファイルは...

- ダブルクリックすると html ファイルが開かれ web ページが立ち上がる。
- 右クリックして「Visual Studio Code」で開くを選択すると、コードを編集できる。

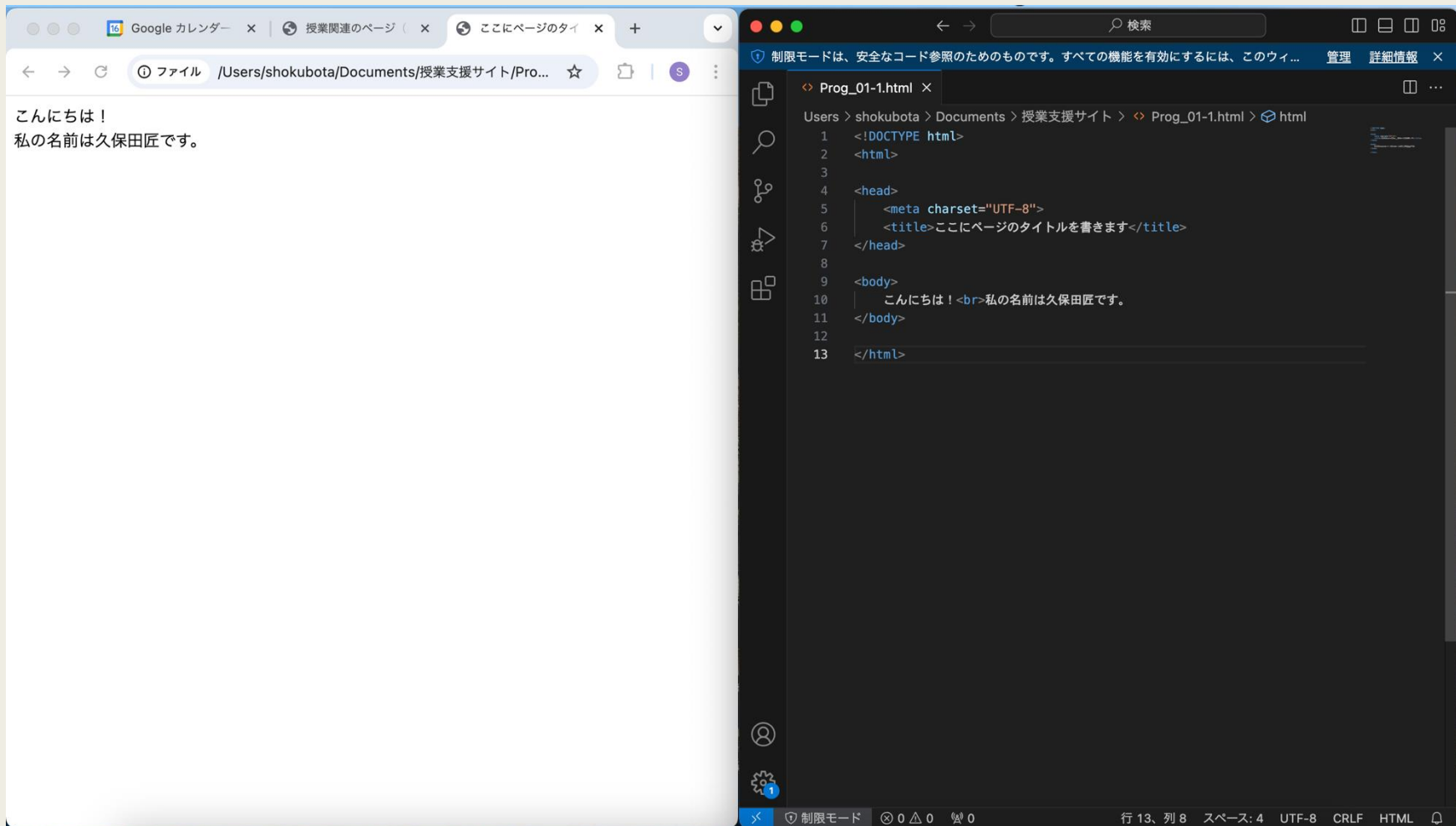


ダブルクリックで「開く」  
右クリックで「編集」

# [準備]作業環境の準備



- 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。



# [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。

```
Prog_01-1.html ×
Users > shokubota > Documents > 授業支援サイト > Prog_01-1.html > html
1  <!DOCTYPE html>
2  <html>
3
4  <head>
5      <meta charset="UTF-8">
6      <title>ここにページのタイトルを書きます</title>
7  </head>
8
9  <body>
10     こんにちは！<br>
11     私の名前は久保田匠です。
12 </body>
13
14 </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、見栄えの観点からは物足りないことが多い。
- CSS は Web ページの見た目を記述する。
  - CSS は Cascading Style Sheets の略。
- ただし、HTML のタグのみでもある程度見栄えを整えることはできる。

```
<> Prog_03-1.html ×
Users > shokubota > Documents > 授業支援サイト > <> Prog_
1  <!DOCTYPE html>
2  <html>
3
4  <head>
5    <meta charset="UTF-8">
6    <title>Prog_03-1</title>
7  </head>
8
9  <body>
10   普通のテキスト
11   <center>中央寄せ</center>
12   <font color="red">赤色のテキスト</font>
13   <br>
14   <font size="6">大きいテキスト</font>
15 </body>
16
17 </html>
```



# [復習] style 属性を利用してcssを使う

- 最も簡単な方法は、HTMLのタグ内で **style 属性**を使う方法。
- ただし、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。
- 基本的にはこれができれば CSS については問題なし。

style 属性

```
Prog_03-2.html x
Users > shokubota > Documents > 授業支援サイト > Prog_03-2.html > html
1  <!DOCTYPE html>
2  <html>
3
4  <head>
5    <meta charset="UTF-8">
6    <title>Prog_03-2</title>
7  </head>
8
9  <body style="text-align: center">
10   <p style="color: red">赤色のテキスト</p>
11   <p style="color: green; font-size: 24px;">緑色, 24px</p>
12   <p>
13     テキストの一部にスタイルを適用する場合は、
14     <span style="text-decoration: underline">span要素</span>を使う。
15   </p>
16 </body>
17
18 </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行目のように複数使う場合はセミコロン（;）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、テキストの装飾的な線を表示	text-decoration: underline;

# 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)

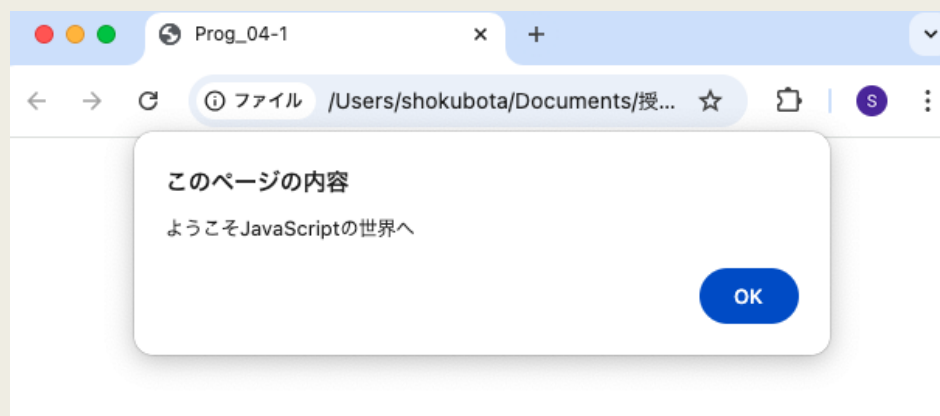
```
<body>
<p>
  <script>
    var point = prompt("試験の点数を0から100以下の整数値で入力してください。");
    document.write("あなたの成績は" + point + "点でした。")
    if(point >= 60){
      document.write("単位取得おめでとう!")
    } else {
      document.write("不合格です。また来年頑張ってください。")
    }
  </script>
</p>
</body>
```

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 は Web ページに「動き」を与える。例えば...
  - 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。

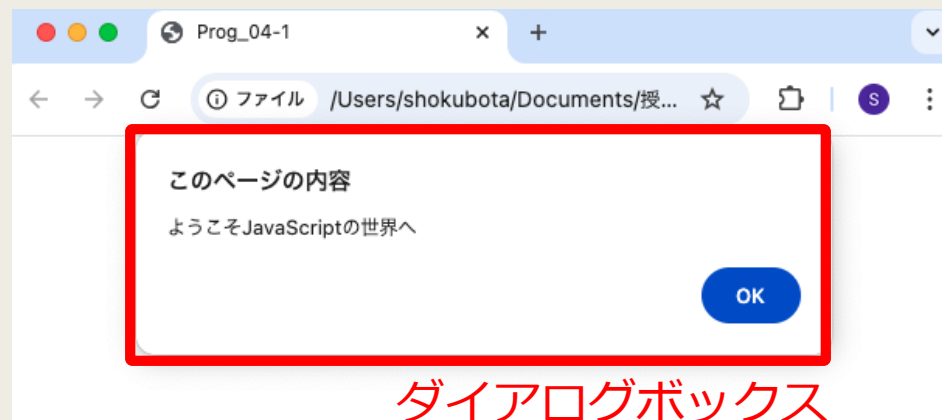
```
1  <!DOCTYPE html>
2  <html>
3
4  <head>
5      <meta charset="UTF-8">
6      <title>Prog_04-1</title>
7  </head>
8
9  <body>
10     <script>
11         alert("ようこそJavaScriptの世界へ");
12     </script>
13 </body>
14
15 </html>
```



# 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。
- 文字列を出力させる場合はダブルクォート(“)と(”)で囲む。
- ステートメントの終わりにはセミコロン ( ; ) をつけるクセをつけよう (JavaScriptではコード内で適切に改行すればセミコロンは省略可) 。

```
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>
```



# 簡単な計算を試みよう

- 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>
```

このページの内容

9

OK

- 「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ォートで囲まれているが、2行目はダブルクォ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回の資料参照）。
- 同様の機能が JavaScript にもあり、`<script>` タグ内で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 `*/` で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>
```

このページの内容

1 + 1 = 2

OK

- 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。

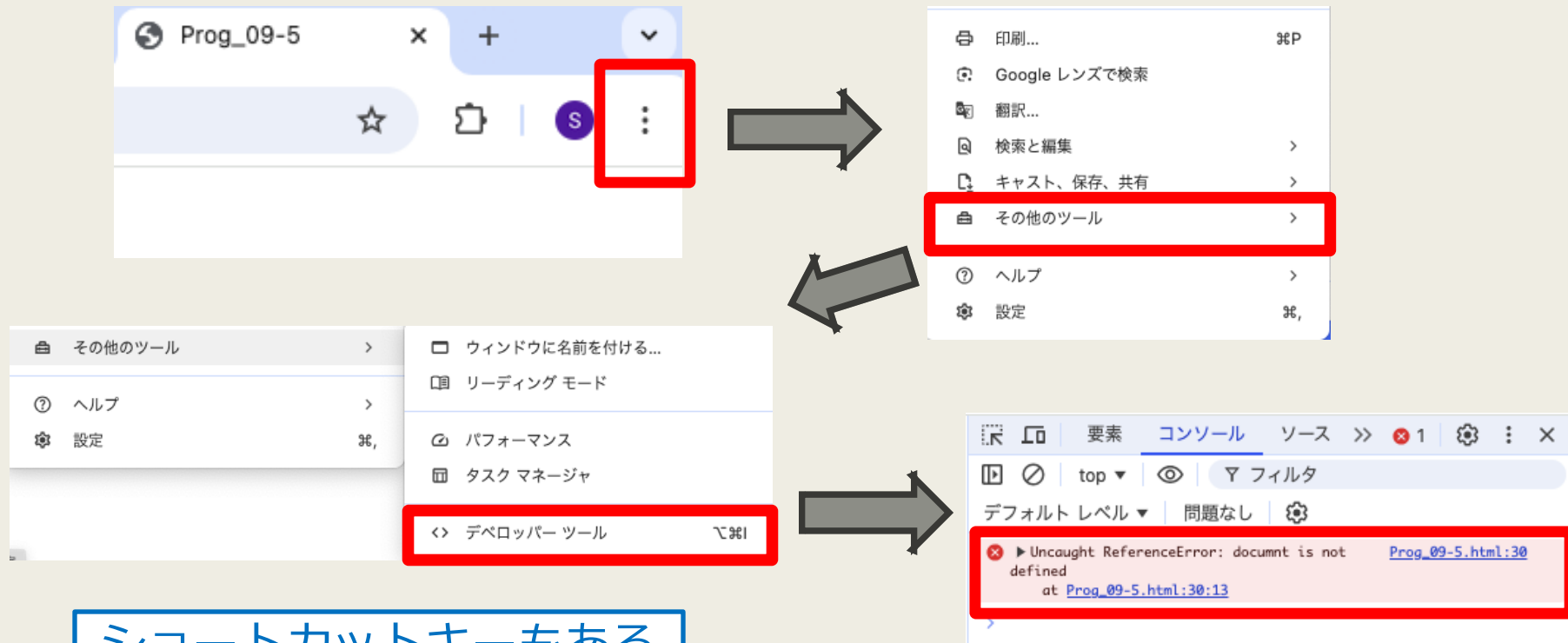
```
1  <!DOCTYPE html>
2  <html>
3
4  <head>
5      <meta charset="UTF-8">
6      <title>Prog_04-1</title>
7  </head>
8
9  <body>
10     <h1>ようこそJavaScriptの世界へ</h1>
11     <p>
12         計算結果は
13         <script>
14             document.write(12 + 7);
15         </script>
16         です。
17     </p>
18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

- 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を見てみよう。



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。  
documnt 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。
- 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JavaScript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JavaScript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

- 文字列同士の足し算 → 連結

```
alert(“やき” + “にく”); //やきにく
```

↑JavaScript におけるコメントアウト

- 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”)
```



# [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。