

# SAPPORO CAMP Day01

---

html, css, JavaScript



# Who am I ??



@taro\_osg

Taro Osg (33)

G's ACADEMY FUKUOKA 主任講師

茨城 -> 札幌(2006 - 2013) -> 東京 -> 福岡

JavaScript / React / Node.js

PHP / Laravel / Rust



# CONTENTS

---

- プログラミングの考え方
- HTML
- CSS
- JavaScript

# プログラミングの考え方



「わかる」->「できる」



「できる」->「わかる」

①「よくわからんけど動いた！」

②「前書いたのと同じように！」

③「そういうことだったのか！」



「わかる」->「できる」

✓ 「できる」->「わかる」



「わかる」->「できる」



「できる」->「わかる」

「できた」をどれだけ積み重ねるか

「コード」=「量」

とにかく書こう！！

# もう少し詳しく. . ?

---

プログラミングってなにがいいの？？

- <https://taroosg.io/why-programming-is-interesting>

チャレンジするときのコツとか？？

- <https://taroosg.io/the-idea-of-how-to-achieve-it>

- 本日のゴール -

自作した完成品が手元にある！

自分の言葉でメモを残す！

自分で作れそうな感覚を得る！

# 本日実装するアプリケーション

おみくじアプリ！！

# 運試しおみくじ

クリックで  
結果表示！



結果は？？

1日でつくります(｀・ω・)b

 注意するポイント 

打ち間違い

# 打ち間違い

打ち間違い

# HTML / CSS / JavaScriptの役割

# HTML, CSS, JavaScriptの役割

---

- HTML
  - 「何を表示するか」を決める.
    - webアプリケーションのコンテンツを指定する.
    - 文章 / 画像 / 音声 / 動画 / etc...
- CSS
  - 「どうやって表示するか」を決める
    - webアプリケーションの装飾を行う.
    - 文字の大きさ / 色 / 背景 / 画像の大きさ / レイアウト / etc...
- JavaScript
  - 「アプリケーションの操作」を決める.
    - クリック時の操作 / 表示の切り替え / 外部からのデータ取得 / etc...

それぞれ役割が異なる．．！

# HTML

# 運試しおみくじ

ボタンが4つ！



ボタンには  
画像と文字！

結果は？？

# HTMLだけだとこんな感じ. . !

---

## 運試しおみくじ



健康運



恋愛運



経済運



学業運

結果は？？

# HTML全体の構造(決まっている)

---

```
<!DOCTYPE html>
<html lang="ja">

<head> // 文書自体の情報（画面には表示されない）
  <meta charset="UTF-8">
  <title>Document</title>
</head>

<body> // ブラウザの画面に表示される部分
  ここにコンテンツを記述する。
</body>

</html>
```

# HTMLはタグを使う(たくさんある)

---

- タグは<開始タグ>と</終了タグ>がある

- 例 :

```
<section>文書の区切りを表すsectionタグ</section>
<h1>タイトルを記述するh1タグ</h1>
<p>文を記述するpタグ</p>
<button>ボタンをつくるbuttonタグ</button>
```

- 1つのタグで完結するものもある

- 例 :

```
 (画像を表示するimgタグ)
```

# HTMLの始め方

---

- 全体の構造をつくる
  - エディタで「!」を入力する(半角で入力すること).
  - 「tab」キーを入力する.
  - 全体の構造ができる！

```
1  <!DOCTYPE html>
2  <html lang="en">
3
4  <head>
5    <meta charset="UTF-8">
6    <meta name="viewport" content="width=device-width, initial-scale=1.0">
7    <title>Document</title>
8  </head>
9
10 <body>
11
12 </body>
13
14 </html>
```

# まずは書いてみよう！！

---

```
<!DOCTYPE html>
<html lang="ja">

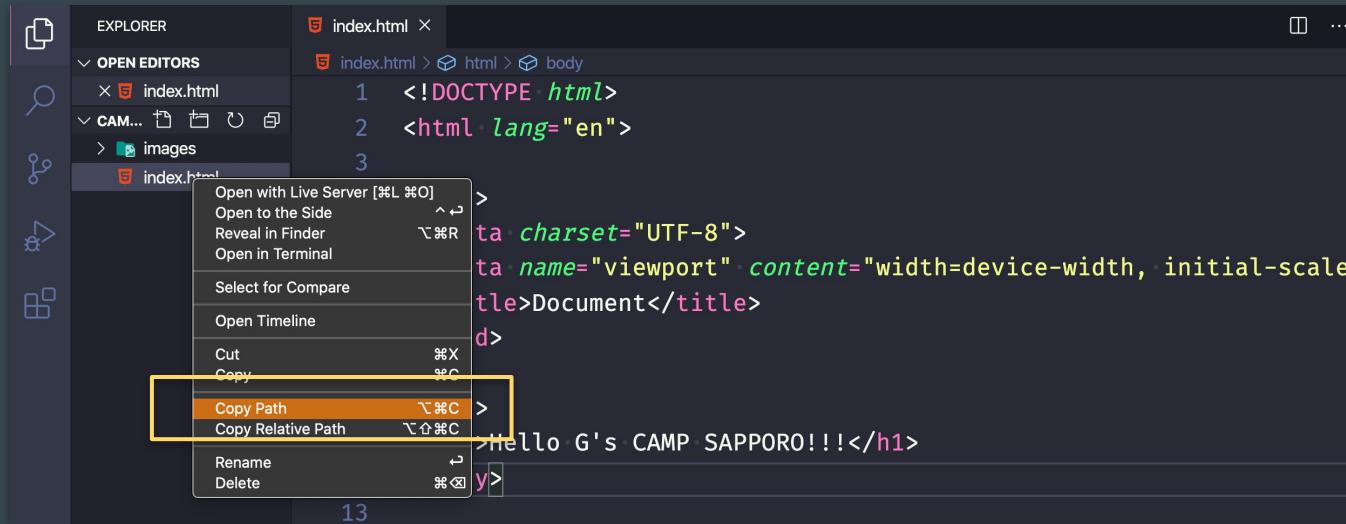
<head>
  <meta charset="UTF-8">
  <title>Document</title>
</head>

<body> // 下の1行を追記しよう！
  <h1>Hello G's CAMP SAPPORO!!!</h1>
</body>

</html>
```

# 書いたHTMLを動かそう！①

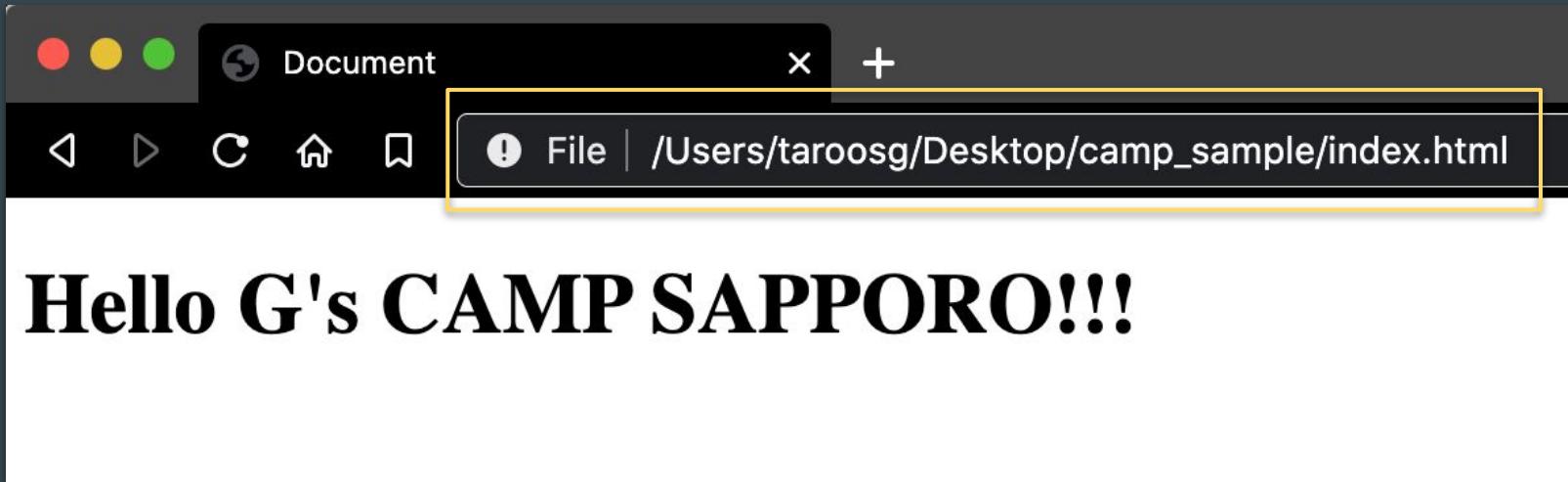
- ブラウザで動かす
  - エディタで動かしたいファイルを右クリック.
  - 「Copy Path」を選択する.



## 書いたHTMLを動かそう！②

---

- ブラウザで動かす
  - ブラウザのアドレスバーに貼り付けして「Enter」.
  - 下のように表示されればOK！



# おみくじのHTMLを書く！

# 運試しおみくじ

タイトルゾーン  
タイトル文字

ボタンゾーン  
ボタンが4つ！



健康運



恋愛運



経済運



学業運

ボタンには  
画像と文字！

結果ゾーン  
結果を表示！

結果は？？

# sectionで区切りを作つて内容を書こう！

---

```
<body> // bodyタグの中に追記しよう！
<main>
  <section>
    <h1>運試しおみくじ</h1>
  </section>
  <section>
    <button>
      健康運
    </button>
  </section>
  <section>
    <p>結果は？？</p>
  </section>
</main>
</body>
```

動作確認(ブラウザで下記のように表示されればOK！)

---

運試しおみくじ

健康運

結果は？？

# ボタンに画像を入れよう！

---

```
<!-- 省略 -->
<section>
  <!-- 「このファイルから見てどの画像ファイルを表示するか」を書く！ -->
  <button>
    
    健康運
  </button>
</section>
<!-- 省略 -->
```

# 動作確認(画像のサイズは大きくてOK！)

---



# ボタンを量産しよう！

---

```
<!-- 省略 -->
<section>
  <button>
    
    健康運
  </button>
  <button>
    
    恋愛運
  </button>
  <!-- 経済運、学業運も記述しよう！ -->
</section>
<!-- 省略 -->
```

# 動作確認(HTMLはこれで完了！)

---

運試しおみくじ



健康運



恋愛運



経済運



学業運

結果は？？

# CSS

# 運試しおみくじ

ボタンサイズ  
枠線の色



健康運



恋愛運



経済運



学業運

文字の大きさ

結果は？？

背景の色  
背景写真

# まずはCSSファイル(とフォント)を読み込む

---

```
<head>
  <meta charset="UTF-8">
  <!-- フォントの読み込み -->
  <link href="https://fonts.googleapis.com/css?family=Noto+Sans+JP"
rel="stylesheet">
  <!-- cssファイルの読み込み -->
  <link rel="stylesheet" href="css/style.css">
  <title>Document</title>
</head>
```

# CSSの書き方(CSSファイルに記述する)

---

```
/* タグに指示するやり方 */
/* 指示する対象（ここではh1タグ） */
h1{
    /* 指示する内容（文字の色を赤に！） */
    color: red;
}
```

```
/* 「クラス」を指定して指示するやり方（好きな場所に指示できる） */
.title{
    font-size: '24px';
}
```

# ボタンをつくる！

コレ！！



# ボタンをつくろう！①

---

```
/* <button>全体に指示を出す！ */
button{
    /* ボタンの大きさ（単位はpx） */
    width: 150px;
    height: 150px;
    /* ボタンの角を丸くする */
    border-radius: 15px;
    /* ボタンの枠線を設定 */
    border: 3px solid #dedede;
    /* 文字の色（カラーコード）と大きさを設定 */
    color: #dedede;
    font-size: 20px;
}
```



# ボタンをつくろう！①

---

```
/* <button>の中の<img>に指示を出す！ */  
button img{  
    /* ボタンの中の図の大きさを設定 */  
    width: 70px;  
    height: 70px;  
}
```



# ここまででこんな感じ！（レイアウトがイマイチ）

---

運試しおみくじ



健康  
運



恋愛  
運



経済  
運



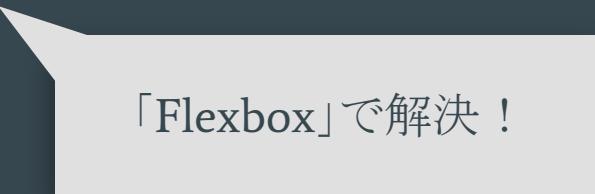
学業  
運

結果は？？

# 【重要】よくあるレイアウト

---

- 横並び
  - HTMLは縦並びがデフォルト.
- 左右中央
  - HTMLは左揃えがデフォルト.
- 上下中央
  - HTMLは上揃えがデフォルト.



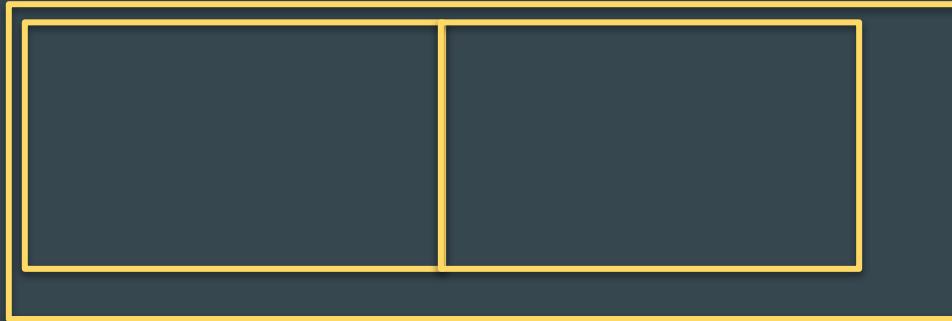
「Flexbox」で解決！

# Flexboxで横並び！

---

- `display:flex`
  - HTMLは縦並びがデフォルト.
  - `display: flex;`を使用すると横並びにできる！(外側の要素に記述！)

`display:flex;`

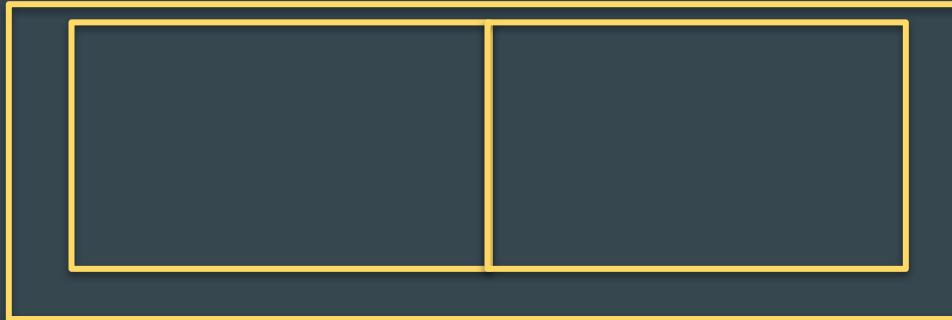


# Flexboxで横並び + 左右中央！

---

- justify-content: center;
  - HTMLは左揃えがデフォルト.
  - justify-contentを設定することで左右を制御できる！

display:flex; justify-content:center;

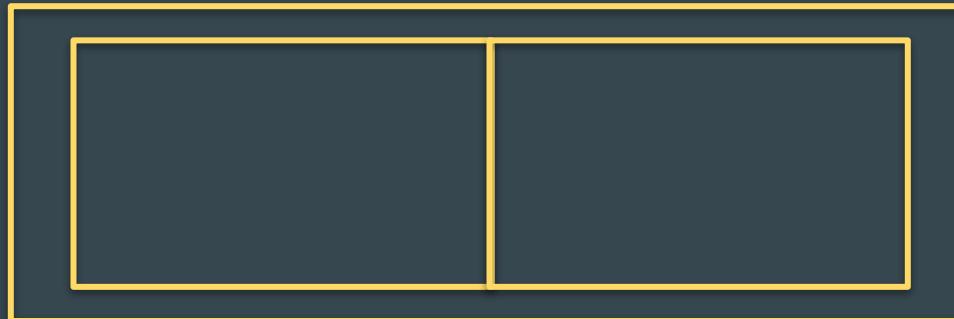


# Flexboxで横並び + 上下左右中央！

---

- align-items: center;
  - HTMLは上揃えがデフォルト.
  - align-itemsを設定することで左右を制御できる！

display:flex; justify-content:center; align-items:center;

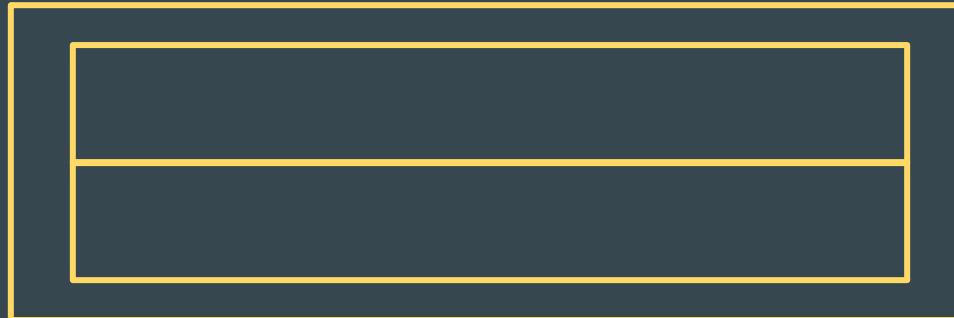


# Flexboxで縦並び + 上下左右中央！

---

- flex-direction: column;
  - Flexboxは横並びがデフォルト.
  - flex-direction: column;を設定することで縦並びにできる！

```
display:flex; justify-content:center; align-items:center; flex-direction:column;
```



## ボタンをつくろう！②

---

```
/* <button>全体に指示を出す！ */
button{
    /* 先程の続きを記述しよう */
    /* ボタンの中の図と文字を縦並び上下左右中央に設定 */
    display: flex;
    flex-direction: column;
    justify-content: center;
    align-items: center;
    /* ボタン周りの余白 */
    margin: 8px;
}
```



# ここまででこんな感じ！（色以外はいい感じ）

---



# ボタンをつくろう！③

```
<!-- 4つあるボタンに「クラス」をつけよう -->
<button class="button_health">
  
  健康運
</button>
<button class="button_love">
  
  恋愛運
</button>
<!-- 他2つも同様に（右表参照） -->
```

ボタン	クラス名
健康運	button_health
恋愛運	button_love
経済運	button_economy
学業運	button_study

# ボタンをつくろう！③

```
/* それぞれのクラスを色を指示するcssを書こう */
/* backgroundで背景の色を設定できる！ */
.button_health{
    background: #a03ead;
}

.button_love{
    background: #ad483e;
}

/* 他2つも同様に（右表参照） */
```

クラス名	色
button_health	#a03ead
button_love	#ad483e
button_economy	#8ead3e
button_study	#3ead74

# ここまででこんな感じ！(ボタンは完成！)

---



# 全体のレイアウトをつくる！

# 運試しおみくじ



健康運



恋愛運



経済運



学業運

結果は？？

全体の  
レイアウト

# 全体の大きさや背景を設定しよう①

---

```
/* <main>全体に指示 */
main{
    /* おみくじ全体の大きさを設定 */
    max-width: 480px;
    height: 100vh;
    /* 背景の色を設定 */
    background: #3e62ad;
    /* 文字の色を設定 */
    color: #dedede;
    /* ページの中心に配置 */
    margin: 0 auto;
}
```

# 全体の大きさや背景を設定しよう①

---

```
/* <main>全体に指示 */
main{
    /* 前ページの続き */
    /* レイアウトの設定 */
    display: flex;
    flex-direction: column;
    justify-content: space-evenly;
}

/* space-evenlyは余白が均等になるよう配置してくれる！ */
/* 「flexbox チートシート」でググろう！ */
```

# ここまででこんな感じ！（全体のレイアウトは完成！）

---



## 全体の大きさや背景を設定しよう②

---

```
<!-- 3つのsectionに「クラス」をつけよう -->
<main>
  <section class="title_section">
    <h1>運試しおみくじ</h1>
  </section>
  <section class="button_section">
    <!-- ボタン部分 -->
  </section>
  <section class="result_section">
    <p id="result">結果は？？</p>
  </section>
</main>
```

## 全体の大きさや背景を設定しよう②

---

```
/* タイトル部分のクラスに指示 */
.title_section{
    /* タイトル文字の大きさ */
    font-size: 32px;
    /* タイトルを上下左右中央に設定 */
    display: flex;
    justify-content: center;
    align-items: center;
}
```

## 全体の大きさや背景を設定しよう②

---

```
/* ボタン部分のクラスに指示 */
.button_section{
    /* ボタンを上下左右中央に配置 */
    display: flex;
    justify-content: center;
    align-items: center;
    /* はみ出したら下に回り込ませる設定 */
    flex-wrap: wrap;
}

/* flex-wrapは要素がはみ出したときの設定 */
/* 「flexbox チートシート」でググろう！ */
```

## 全体の大きさや背景を設定しよう②

---

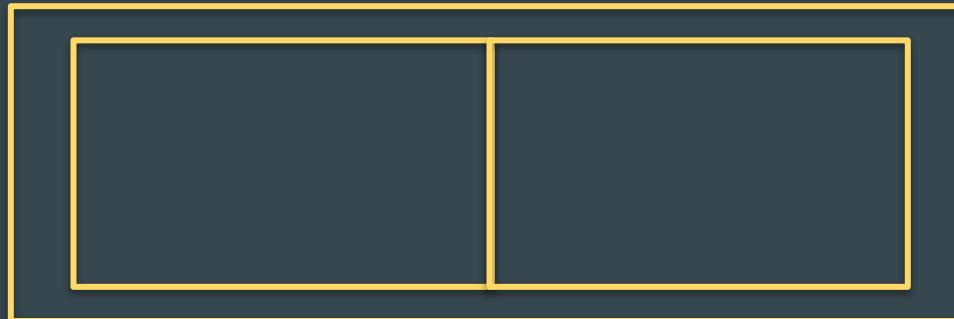
```
/* 結果表示部分のクラスに指示 */
.result_section{
    /* 結果の文字の大きさ */
    font-size: 48px;
    /* 上下左右中央に配置 */
    display: flex;
    justify-content: center;
    align-items: center;
}
```

# Flexboxで横並び + 上下左右中央！ のパターン

---

- align-items: center;
  - HTMLは上揃えがデフォルト.
  - align-itemsを設定することで左右を制御できる！

display:flex; justify-content:center; align-items:center;



# ここまででこんな感じ！（できた感が出てきた！）

---



背景に写真を入れよう！

# 運試しおみくじ



健康運



恋愛運



経済運



学業運

結果は？？

背景に写真  
ぼやかす！

# ページ全体の背景を写真にする！

---

```
/* <body>全体に指示 */
body{
    /* ↓↓↓ 最初からあるやつ ↓↓↓ */
    margin: 0;
    font-family: "Noto Sans JP";
    /* 背景画像の写真を設定（画像を指定する） */
    background: url('../images/main.JPG');
    /* 背景画像のサイズと位置を設定 */
    background-size: cover;
    background-position: center;
}
```

# 写真が入った！！！

---



## 運試しおみくじ



結果は？？



# ぼかす！

---

```
/* <body>全体に指示 */
body{
    /* ↓↓↓ 最初からあるやつ ↓↓↓ */
    margin: 0;
    font-family: "Noto Sans JP";
    /* 背景画像の写真を設定（画像を指定する） */
    background: url('../images/main.JPG');
    /* 背景画像のサイズと位置を設定 */
    background-size: cover;
    background-position: center;
    /* 背景をぼかす */
    backdrop-filter: blur(10px);
}
```

WRYYYY!!!!!(CSS完成ッ！！！)

---

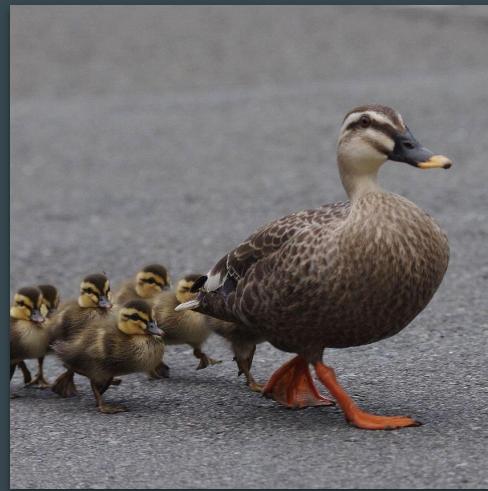


# JavaScript

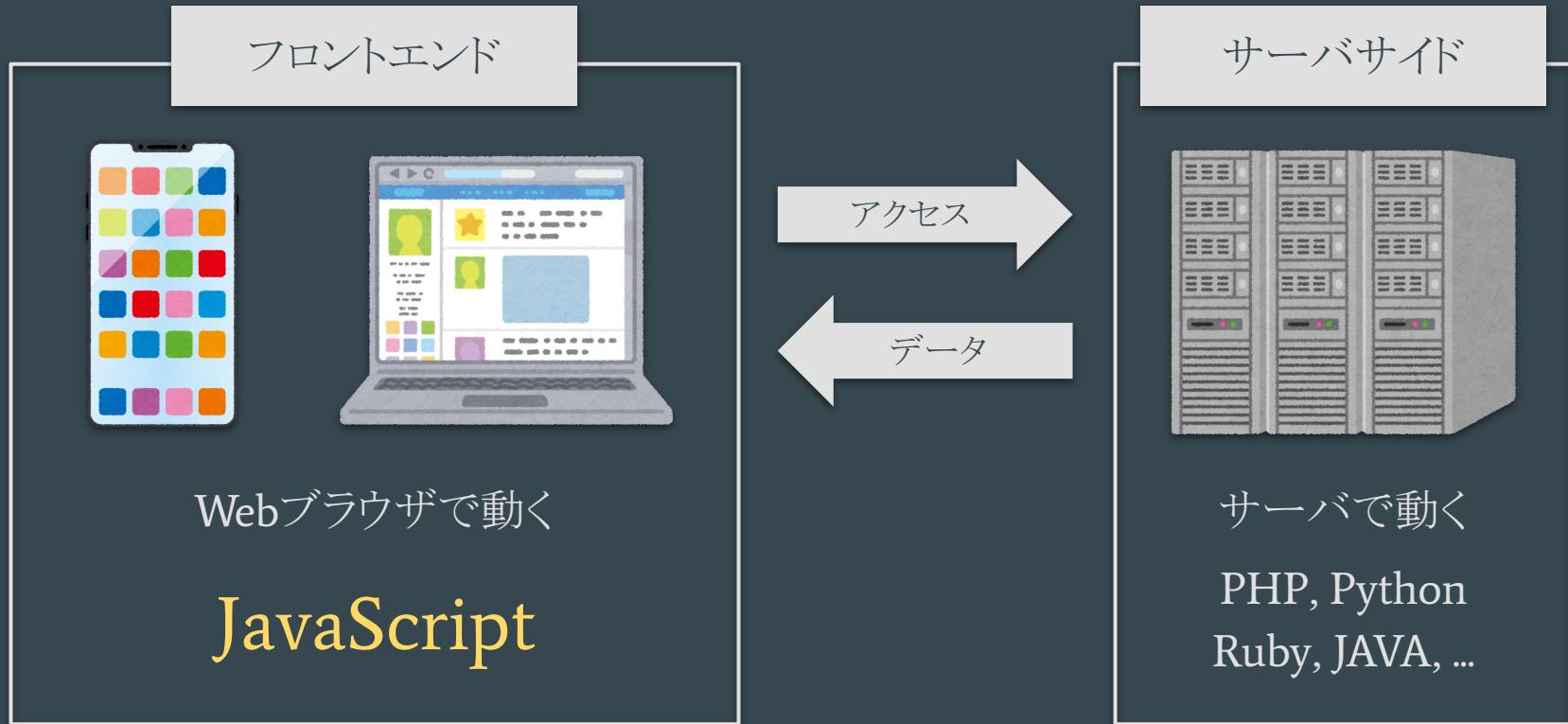
# JavaScript ≠ JAVA



≠

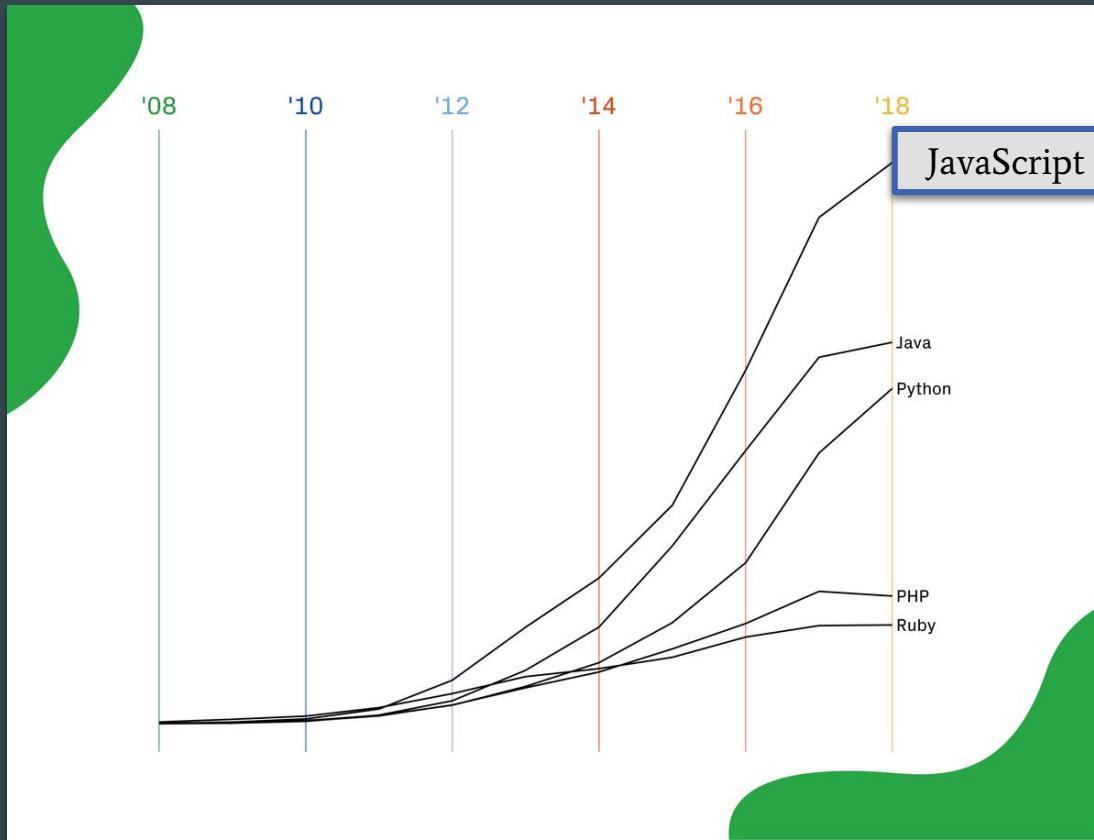


# Webアプリケーションの構造

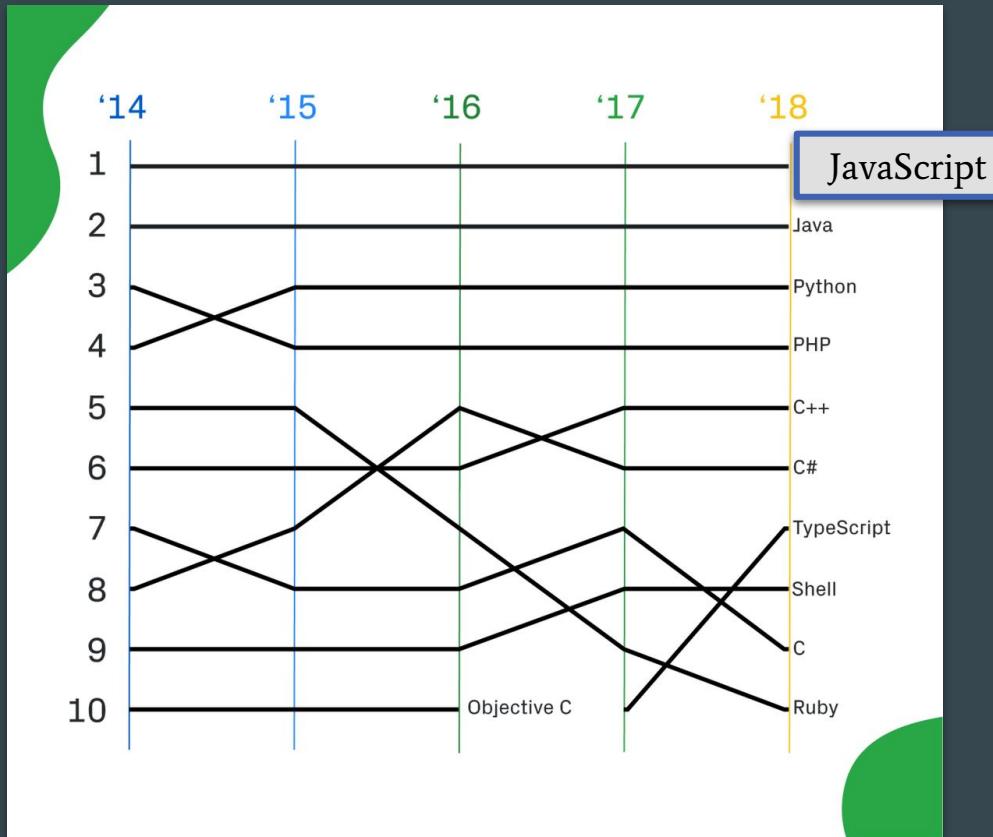


# JavaScriptはwebアプリに欠かせない！

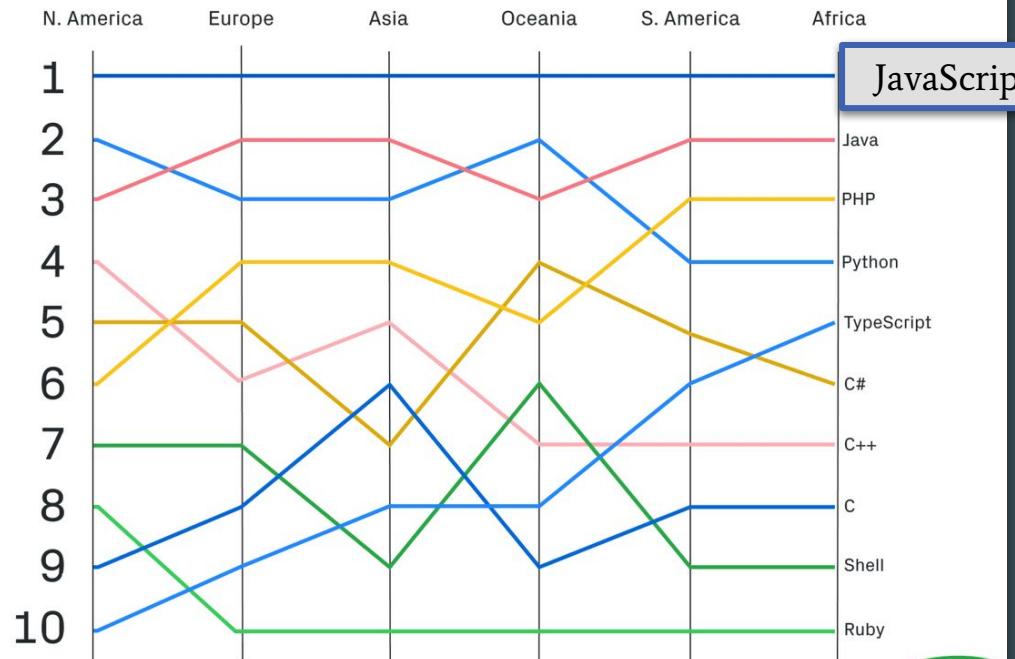
# Githubのリポジトリ数



# 言語別人気ランキング



# 地域別人気ランキング



# JavaScript周辺の技術



# よく使われる技術の紹介

---



- JavaScriptのライブラリ.
- ユーザ操作イベントやアニメーションを実現
- ✓ 生JSと比較して短くかける.
- ✓ webアプリケーションで広く普及している.
- ✓ 導入が簡単.
- ✓ 学習コストが低い.
- ▲ 難しいことをやろうとすると複雑になる.
- まずはここから !

# よく使われる技術の紹介

---



- JavaScriptのライブラリ.
- ✓ モダンなwebアプリケーション(SPA)を実現
- ✓ 高速！
- ✓ スマホアプリも見据えた開発が可能.
- ▲ 学習コスト / 環境構築がややハードル.
- (私はReactが好き)

# よく使われる技術の紹介

---



- サーバサイドでJavaScriptを動かす技術.
- ✓ フロントもサーバもJavaScriptで書ける... !
- ▲ 難しいことをやろうとすると複雑になる.

# JavaScriptを書く！

# 早速書く！

---

- 書き方
  - <script></script>の間に処理を記述
- 書く場所
  - htmlファイルの</body>のすぐ上に書こう！
  - ほかにもいくつか書ける場所があります.
  - 別にファイルを作るやり方もあります.

# 早速書く！

---

```
<!DOCTYPE html>
<html lang="ja">
<head>
    <!-- 省略 -->
</head>
<body>
    <!-- 他のHTML要素 -->
    <script>
        // ここにJavaScriptを書く！！！
    </script>
</body>
</html>
```

# 早速書く！

---

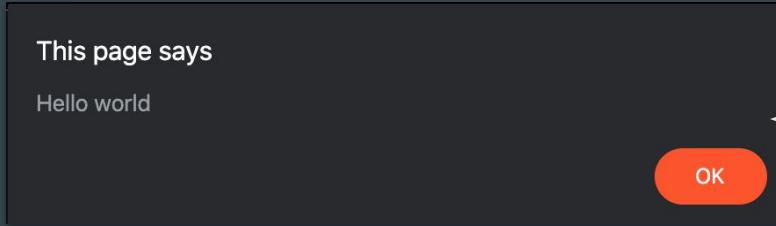
```
<script>
  // alert();でポップアップ表示！
  alert('Hello world');    // 文字列は「'」か「"」で囲む.

  // console.log(); ブラウザで検証ツール→consoleで確認！
  console.log('Hello world');
</script>
```

# 早速書く！

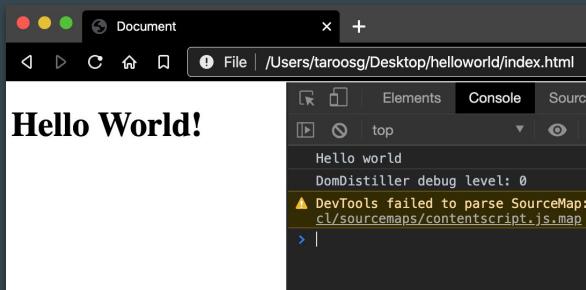
---

- 前ページを参考にalert()とcnssole.log()を動かそう！
  - alert();



リロードすると表示される！

- console.log();

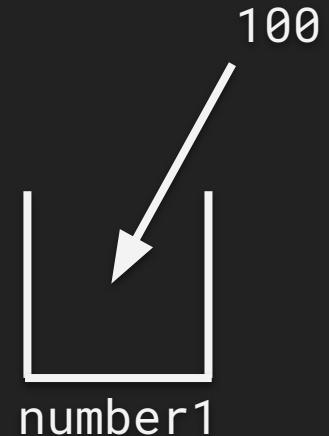


consoleに表示される！

# 変数と計算

# 変数とは？？

```
<script>  
  
// 「変数」は文字列や数値を入れるための箱のようなもの  
// 箱に名前をつけておき、あとから再度利用できる！  
  
const text1 = 'hello';           // 文字列（' 'などで囲む）  
const text2 = 'world';          // 文字列  
const number1 = 100;            // 数値  
const number2 = 200;            // 数値  
const 3number = 1000;           // 名前の先頭が数字はダメ  
  
</script>
```



# 計算

---

```
<script>  
    // 数学と同様に計算できる！  
  
    const number1 = 1 + 9;          // 10  
    const number2 = 1 - 5;          // -4  
    const number3 = 2 * 4;          // 8  
    const number4 = 10 / 2;         // 5  
    const number5 = 10 % 3;         // 1  
</script>
```

# 計算

---

```
<script>

// 文字列は結合される

const number1 = 100;                      // 数値
const number2 = 200;                      // 数値
const text1 = "hello";                     // 文字列
const text2 = "world";                     // 文字列
const sum1 = number1 + number2;           // 300 (数値)
const sum2 = text1 + text2;                // helloworld (文字列)

</script>
```

# 乱数(ランダムな数)をつくる！

# 計算

---

```
<script>

// Math.random()を使う！（JavaScriptに最初から用意されている）

const randomNumber1 = Math.random();
alert(randomNumber1);    // 0から1の間でランダムな値（乱数）を表示。

const randomNumber2 = Math.floor(Math.random() * 5);
alert(randomNumber2);    // 0から4までのどれかが表示される！

</script>
```

# 練習！

---

- 以下の数をランダムで発生させてalert();で表示させよう！
  - a. 0から9のどれか
  - b. 1から9のどれか
  - c. 5から10のどれか
  - d. 50から99のどれか

# 条件分岐(if文)

# 計算

---

```
<script>  
    // 条件を満たすときと満たさないときで別々の処理を実行する！  
  
    if(条件式){  
        // 条件式を満たす場合の処理  
    } else {  
        // 条件式を満たさない場合の処理  
    }  
  
</script>
```

# 計算

---

```
<script>

    // 複数の条件で処理を分岐させることもできる！

    if(条件式1){
        // 条件式1を満たす場合の処理
    } else if(条件式2) {
        // 条件式1と満たさなくて条件式2を満たす場合の処理
    } else {
        // いずれの条件も満たさない場合の処理
    }

</script>
```

# 条件式

---

- 条件の書き方(比較演算子)
  - == 等しければtrue
  - != 等しくなければtrue
  - > 左側のほうが大きければtrue
  - < 右側のほうが大きければtrue
  - >= 左側が右側以上ならtrue
  - <= 右側が左側以上ならtrue



# おみくじの処理を作ろう！

---

- ランダムに「大吉・中吉・小吉・凶・大凶」をalert()で表示！
- ヒント！！
  - Math.random()関数で0から4を発生させる。
  - 出た数値に応じてif文を使って条件分岐し、異なる内容をalert()で出力！

# おみくじアプリを実装！

# 運試しおみくじ

クリック時の  
動作



ランダムな  
結果の表示

結果は？？

# おみくじのWebアプリ！

---

- 画面上のHTML要素(DOM)をクリックして処理を実行！
- 要素を「指定」する！
  - classやidでDOMを特定する.
  - 指定したDOMに対してJavaScriptで操作を行う！
- 例
  - 「idがbutton」の要素を「クリック」したら. . .
  - 「大吉-大凶のどれかをランダムに表示」！

# 【参考】DOM

- HTMLに記述されている各要素のこと(document object model)

The screenshot shows a web browser window with the following elements:

- Header:** ログインしていません トーク 投稿記録 アカウント作成 ログイン
- Navigation:** ページ ノート 閲覧 編集 履歴表示 Wikipedia内を検索
- Content:** Document Object Model
- Text:** 出典: フリー百科事典『ウィキペディア (Wikipedia)』
- Text:** Document Object Model (DOM) は、HTML 文書や XML 文書を各種プログラムから利用するための仕組みである<sup>[1]</sup>。WHATWGが Living Standardとして定義している。
- Text:** WHATWG以前は W3C が仕様を策定しており、Level 1 から Level 4 まで勧告している。
- Text:** XML を読み込む API である SAX と異なり、XML
- DOM Inspector:** Elements tab selected. Shows the DOM tree structure of the current page, highlighting the body element.
- Sample HTML:** A small window showing a simplified version of the page's HTML code.

# 【重要】基本の考え方

---

- 基本の3要素



なんだけど...

JavaScriptはDOM操作が苦手. . .



# jQueryライブラリ



# 【重要】最初に読み込みが必要！

```
<script src="https://ajax.googleapis.com/ajax/libs/jquery/3.5.1/jquery.min.js"></script>
<script>
    // jsの処理
</script>
```

# jQueryとは？？

---

- CSSと同じ要領で対象箇所を指定できる
- 素のJavaScriptよりも短く書ける！ <- 重要
- アニメーションなど手軽に設定できる。
- 書き方(順序や考え方)はJavaScriptと同様！ <- 重要
- 導入が簡単(フレームワークなどは環境構築で詰む)

jQueryはJavaScriptを短縮して書けるライブラリ

【参考】[https://webkikaku.co.jp/homepage/blog/hpseisaku/webdesign/jquery\\_start/](https://webkikaku.co.jp/homepage/blog/hpseisaku/webdesign/jquery_start/)

# 【重要】基本の考え方

---

selector  
(どこを)



event  
(いつ)



method  
(どうする)

```
<script>
  $('#button').on('click', function () {
    alert('Hello World!');
  });
</script>
```

# 【重要】基本の考え方

---

selector  
(どこを)



event  
(いつ)



method  
(どうする)

```
<script>
$(セレクタ名).on(イベント名, function () {
    実行したい処理（メソッド）
});
</script>
```

# セレクタ / イベントなど

---

- たくさんあります
  - 「jQuery セレクタ」でググる！
  - 「jQuery イベント」でググる！

# まずは形の入力に慣れよう！

```
$( '#id' ).on( 'click' , function () {  
    // ...  
});
```

(口に出しながら書くと定着する)

だらーあいでいーおんくりっくふあんくしょんかっこかっこなみかっこえたー...

# おみくじの処理を作ろう！①

---

```
<!-- HTML要素にidをつけよう！ -->
<!-- まずは健康運と結果の部分だけでOK！ -->

<button class="button_health" id="pull_health">
  
  健康運
</button>

// ...省略

<p id="result">結果は？？</p>
```

## おみくじの処理を作ろう！②

---

```
<script>
  // 「ボタンクリック時の」処理
  // まず「ボタンクリックイベントが動くかどうか」を確認

  $('#pull_health').on('click', function () {
    alert('OK');
  });
</script>
```

# おみくじの処理を作ろう！③

---

```
<script>
  // 「0から4のランダムな数」の生成
  // できた数によって条件分岐
  $('#pull_health').on('click', function () {
    const randomNumber = Math.floor(Math.random() * 5);
    if (randomNumber == 0) {
      $('#result').text('大吉');    // text()で指定した部分を書き換え
    } else if (randomNumber == 1) {
      $('#result').text('中吉');
    }
    // ...省略
  });
</script>
```

# 運試しおみくじ

「健康運」を  
クリック！



大吉-大凶の  
どれかが表示

結果は？？

# 完成！！！

---

他のボタンも同様につくってみよう！

- それぞれのボタンにidを追加！
- 各idにクリックイベントを作成！
- 「乱数生成 -> 結果表示」の処理を追加！

# 課題

# 課題

---

- 「じゃんけん」を実装！！
  - 自分の手を決めるとコンピュータがランダムな手を出す.
  - 結果に応じて「勝ち」「負け」「あいこ」を表示！
  - アレンジして発展させるのがジーズ流！！！
- アレンジの例
  - 勝率を計算
  - だんだん勝てなくなるじゃんけん
  - 限定じゃんけん
  - じゃんけんで音ゲー
  - ドラクエ

おしまいだああ(`;ω;')

そんなときは. . .

# 写 ★ 経

「写経」とは誰かが作った「動くコード」をひたすら書き写すこと

```
1  <!DOCTYPE html>
2  <html lang="ja">
3
4  <head>
5    <meta charset="UTF-8">
6    <meta name="viewport" content="width=device-width, initial-scale=1.0">
7    <title>Document</title>
8  </head>
9
10 <body>
11   <header>
12     <h1>じゃんけん</h1>
13   </header>
14   <main>
15     <section>
16       <button id="button_gu">グー</button>
17       <button id="button_cho">チョキ</button>
18       <button id="button_pa">パー</button>
19     </section>
20     <section>
21       <p>相手の手は？？</p>
22       <p id="com_hand">ready...</p>
23       <p>結果は？？</p>
24       <p id="result">ready...</p>
25     </section>
```

```
26
27 <script src="https://ajax.googleapis.com/ajax/libs/jquery/3.5.1/jquery.min.js"></script>
28 <script>
29   $('#button_gu').on('click', function() {
30     const randomNumber = Math.floor(Math.random() * 3);
31     if (randomNumber == 0) {
32       $('#com_hand').text('グー');
33       $('#result').text('あいこ');
34     } else if (randomNumber == 1) {
35       $('#com_hand').text('チョキ');
36       $('#result').text('勝ち');
37     } else if (randomNumber == 2) {
38       $('#com_hand').text('パー');
39       $('#result').text('負け');
40     } else {
41       alert('エラー');
42     }
43   });
44 
```

```
44
45  $('#button_cho').on('click', function () {
46    const randomNumber = Math.floor(Math.random() * 3);
47    if (randomNumber == 0) {
48      $('#com_hand').text('グー');
49      $('#result').text('負け');
50    } else if (randomNumber == 1) {
51      $('#com_hand').text('チョキ');
52      $('#result').text('あいこ');
53    } else if (randomNumber == 2) {
54      $('#com_hand').text('パー');
55      $('#result').text('勝ち');
56    } else {
57      alert('エラー');
58    }
59  });
60
```

```
60
61  $('#button_pa').on('click', function() {
62    const randomNumber = Math.floor(Math.random() * 3);
63    if (randomNumber == 0) {
64      $('#com_hand').text('グー');
65      $('#result').text('勝ち');
66    } else if (randomNumber == 1) {
67      $('#com_hand').text('チョキ');
68      $('#result').text('負け');
69    } else if (randomNumber == 2) {
70      $('#com_hand').text('パー');
71      $('#result').text('あいこ');
72    } else {
73      alert('エラー');
74    }
75  })
76
77 </script>
78 </main>
79
80 </body>
81
82 </html>
```

写経！！これでいける！！  
明日の**11:00**まで！！！

# 【おまけ】明日への伏線

# 【補足】配列

```
// 複数の値に順番をつけてまとめて扱う方法。強い。  
// 順番を「index」と呼ぶ。「0」からスタート！
```

// 配列の定義

```
const arr = ['大吉', '中吉', '小吉', '凶', '大凶'];  
console.log(arr[0]); // 大吉
```



# 【補足】オブジェクト

// 配列と同様に複数の値を管理する方法  
// 配列の「index」に対して「key」「value」で管理.

// オブジェクトの定義  
const schedule = {  
 date: '2020/06/21',  
 day: 'Sun',  
 event: 'camp\_day02',  
};

// 値の取得  
console.log(schedule.event); // camp\_day02  
console.log(schedule['event']); // camp\_day02

2020/06/21	Sun	camp_day02
date	day	event

## 【補足】配列とオブジェクトの組み合わせ

```
// オブジェクトが複数あるときは配列と組み合わせる
// APIを使用すると結構出てくる形

// オブジェクトの配列
const schedules = [
  { date: '2020/06/19', day: 'Fri', event: 'meet_up' },
  { date: '2020/06/20', day: 'Sat', event: 'camp_day01' },
  { date: '2020/06/21', day: 'Sun', event: 'camp_day02' },
];

// 値の取得
console.log(schedules[0].date);          // 2020/06/19
console.log(schedules[1].event);          // camp_day01
```

## 【補足】関数(function)

```
// 関数とは記述した処理をまとめて名前をつけて使い回せるようにしたもの。  
// 一度処理を定義してしまえば、呼び出すだけで実行可能！
```

```
// 関数の定義  
function generate0to4() {  
    const randomNumber = Math.floor(Math.random()*5);  
    console.log(randomNumber);  
    return randomNumber;  
}
```

```
// 関数の実行  
const random = generate0to4();  
console.log(random); // 0から4のどれか
```