
Monaca で学ぶ
はじめてのプログラミング

第1章 アプリ開発入門

第1章 アプリ開発入門

Monacaのアカウント登録

1) 教育版公式サイト URL にアクセス

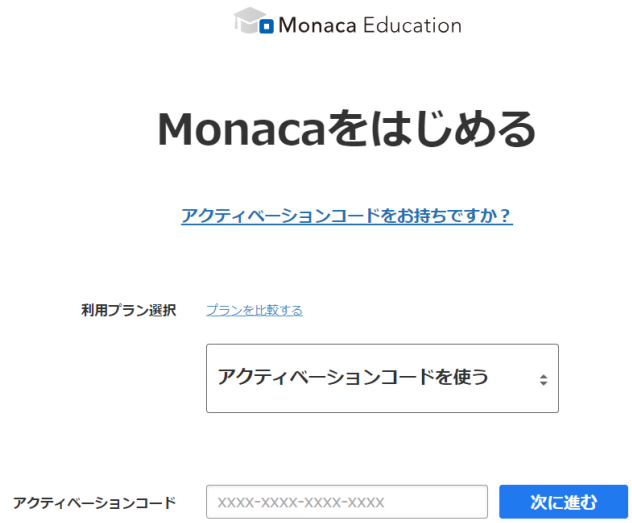
https://edu.monaca.io/

2) 『**アカウント作成**』から移動して必要事項を記入。

3) 仮登録完了メールを確認

4) プラン選択

- Education
  - 有料プランです。利用にはアクティベーションコードが必要です。
- Freeプラン
  - 無料で使えますが容量や機能に制限があります。



第1章 アプリ開発入門

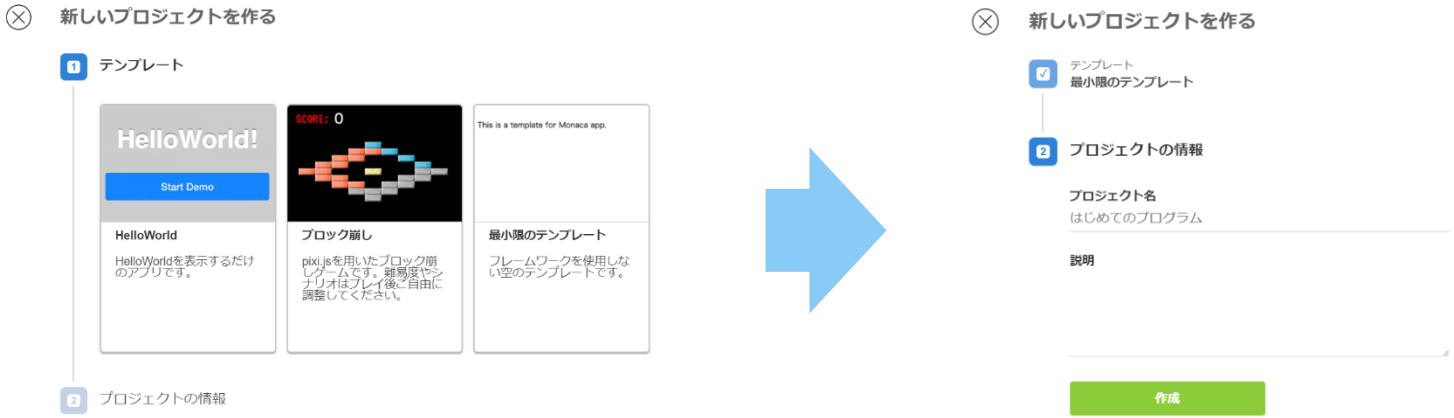
Monaca ダッシュボードとプロジェクト

ログインとダッシュボードの表示

- 開発中のアプリはプロジェクト単位で管理を行う
  - 画面左側に一覧表示される

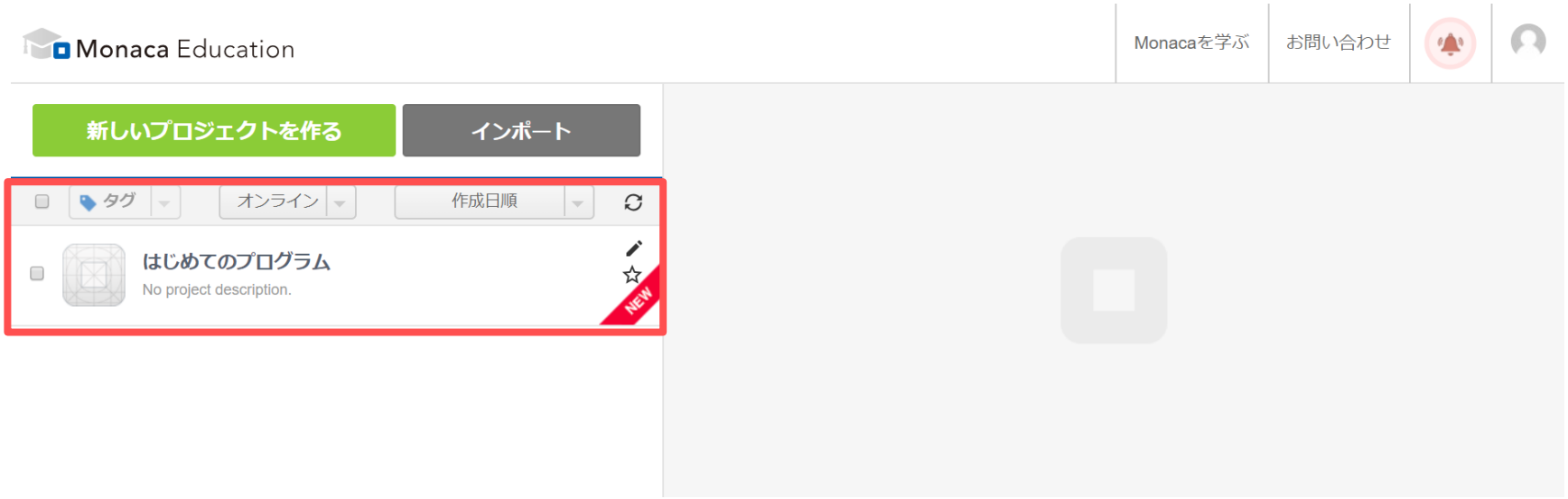
新規プロジェクト作成

1. 『新しいプロジェクトを作る』ボタンをクリック
2. テンプレートから『最小限のテンプレート』を選択
3. プロジェクト名を設定（例：はじめてのプログラム）
4. 『作成』ボタンをクリック



プロジェクト一覧に表示されれば成功

- プロジェクト名
  - 自由に設定可能
  - 後で管理しやすいように気をつける

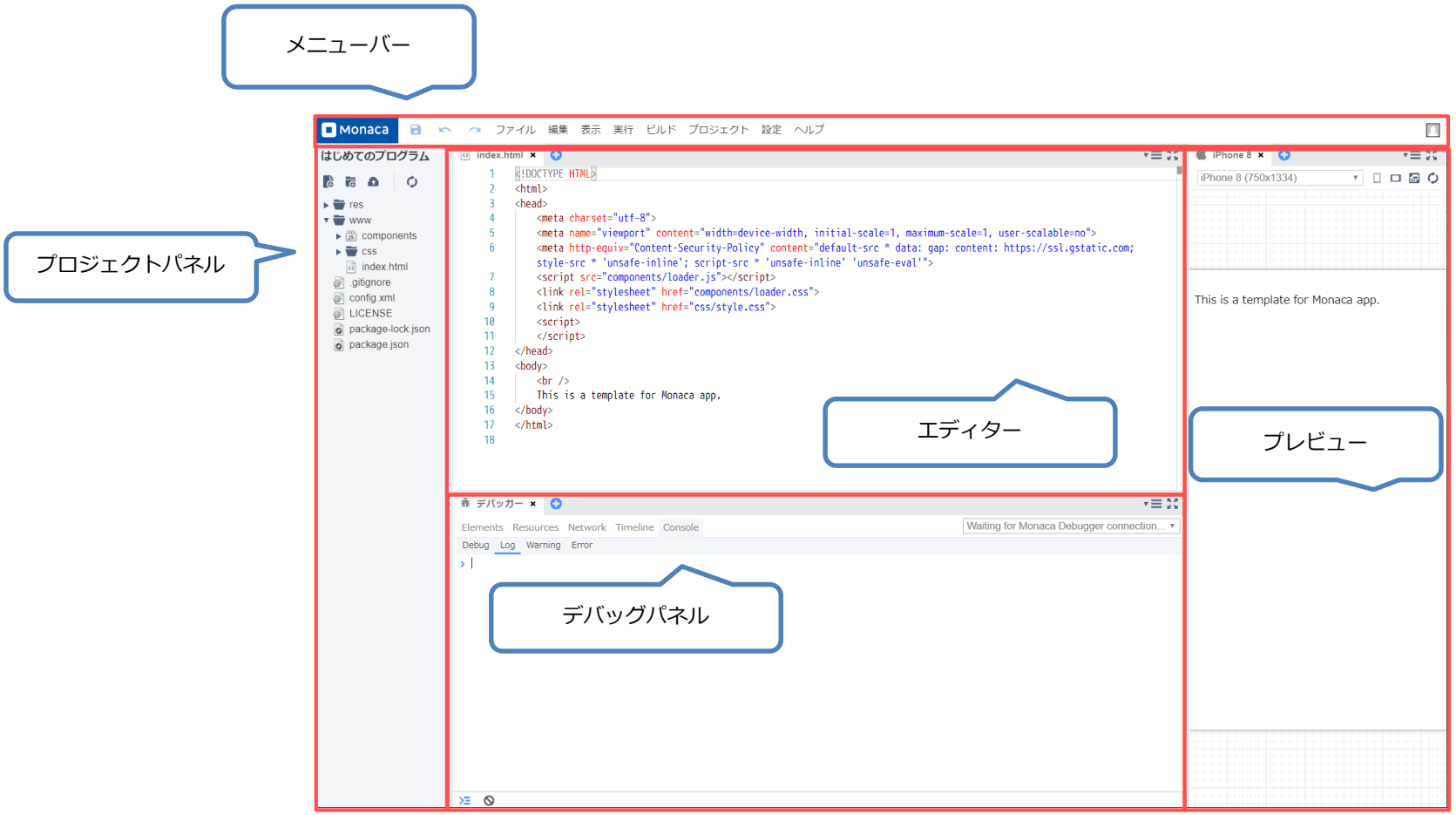


第1章 アプリ開発入門

プログラムを記述する

Monaca IDE

- IDE(Integrated Development Environment)は統合開発環境の意味



エディター

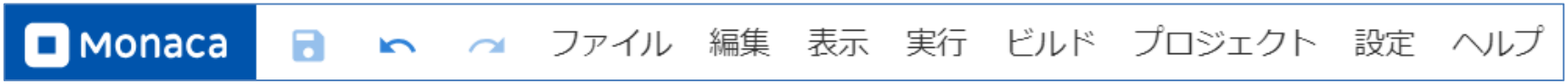
- プログラムを記述するためのパネル

index.html

```
<!DOCTYPE HTML>
<html>
<head>
    <meta charset="utf-8">
    <meta name="viewport" content="width=device-width, initial-scale=1,
    maximum-scale=1, user-scalable=no">
    <meta http-equiv="Content-Security-Policy" content="default-src * data: gap:
    content: https://ssl.gstatic.com; style-src * 'unsafe-inline'; script-src *
    'unsafe-inline' 'unsafe-eval'">
    <script src="components/loader.js"></script>
    <link rel="stylesheet" href="components/loader.css">
    <link rel="stylesheet" href="css/style.css">
    <script>
    </script>
</head>
<body>
    <br />
    This is a template for Monaca app.
</body>
</html>
```

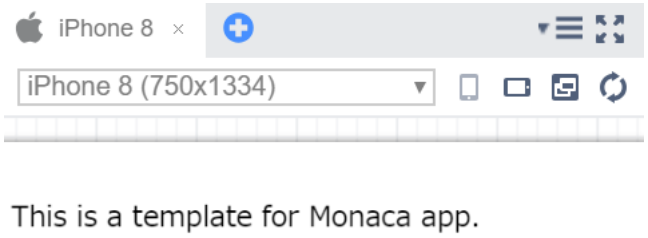
メニューバー

- 各種機能を呼び出す
  - 保存ボタン
  - 元に戻す
  - やり直し
  - ビルドなど



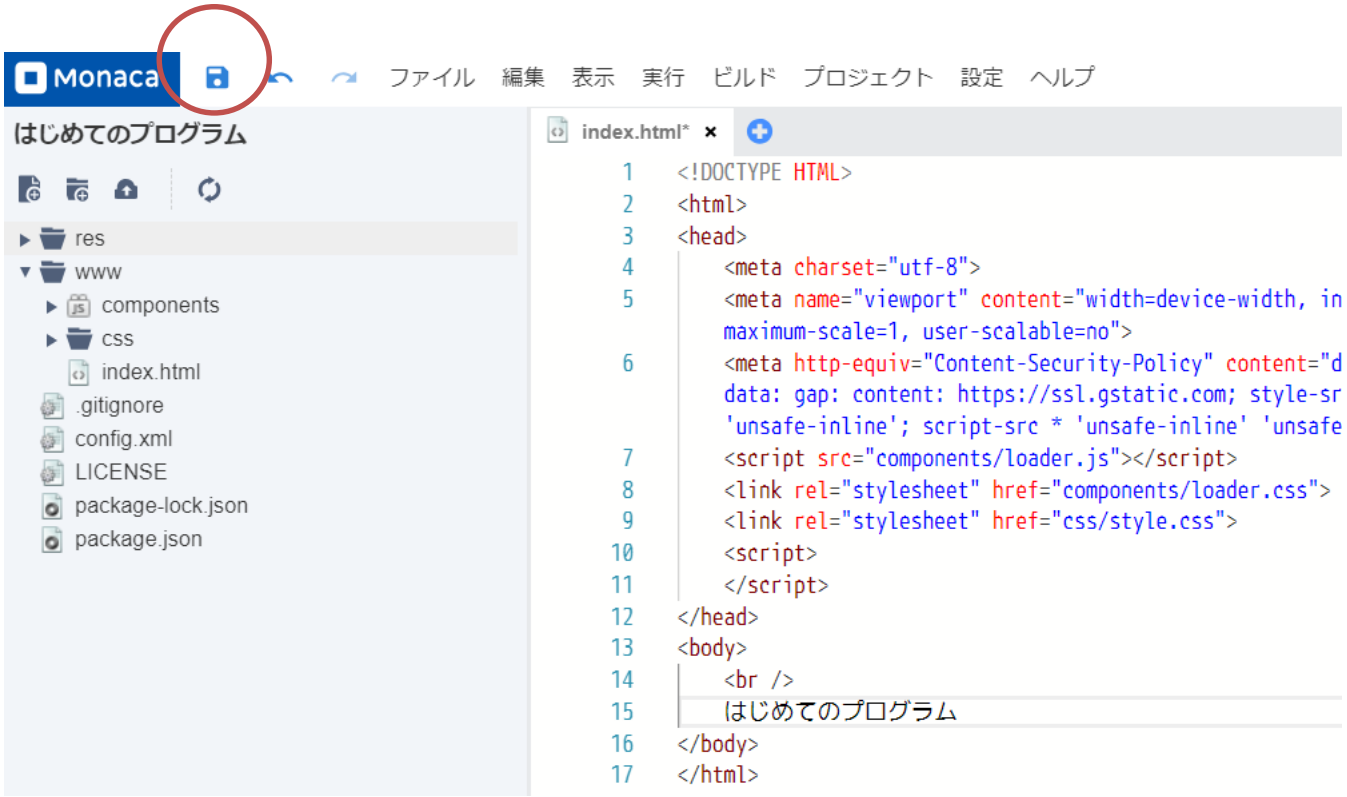
プレビュー

- プログラムの実行結果が表示されます。



はじめてのプログラム

- 「This is a template for Monaca app.」を削除
- 「はじめてのプログラム」と記述
- メニューバーの ［保存］アイコンをクリック



プレビューの更新

- ［保存］と連動してプレビュー画面が自動更新される
- 自動で更新されない場合
  - プレビュー画面右上の円状の矢印ボタンをクリック

JavaScript で簡単な命令を実行させる

- HTML内にJvaScriptを記述する方法
  - <script> で囲まれた部分に記述

```
<script>
  alert("こんにちは");
</script>
```

実行結果

- 次のダイアログが表示される

第1章 アプリ開発入門

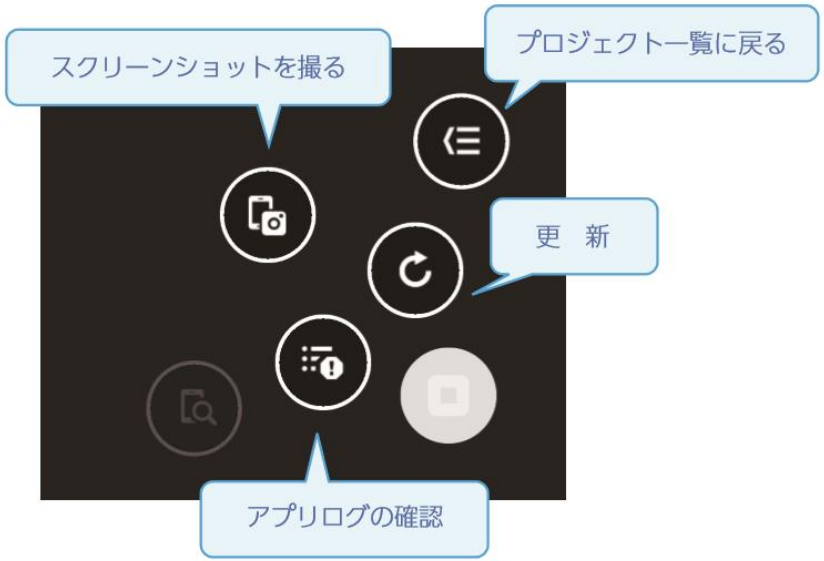
Monaca デバッガーの利用

Monaca デバッガーとは

- アプリの動作を確認するツール
- プレビューより高度な動作確認が可能
  - カメラやコンパスなどのハードウェア機能

Monaca デバッガーのメニュー

- プロジェクトを開きアプリの動作を確認
- 丸いMonacaボタンはメニュー
  - 更新やログの確認が可能



第2章 HTML入門

第2章 HTML入門

HTMLとは

HTML（Hyper Text Markup Language）とは

- マークアップ言語の１つ
- 文書をタグで囲んで記述

HTMLで実現可能なこと

- 文章の構造化
- 文章同士のリンク
- 画像や動画の参照

HTMLタグの書き方

- 「開始タグ」と「終了タグ」にはタグ名が入る
  - タグは全部で100個以上存在する

文法　タグの記述方法と名称

```
<開始タグ>内容</終了タグ>
```

- 例では段落を表すPタグを利用

例　タグの記述例

```
<p>これは段落です。</p>
```

空要素の記述方法

- スラッシュはつけなくても良い
- 例では改行を表すBRタグを利用

文法　空要素の記述方法

```
<開始タグ>
　または
<開始タグ　/>
```

例　空要素の記述例

```
<br>
　または
<br　/>
```

属性について

- タグには属性と属性値を設定できる
- 例ではAタグにhref属性とリンク先の属性値を付加
- 属性値はダブルクォートかシングルクォートで囲む

文法　属性の記述

```
<開始タグ 属性1="値" 属性2="値">内容</終了タグ>
```

例　属性の記述例

```
<a href="top.html">TOPページへ</a>
```

HTMLの例

- HTML文章はタグを入れ子にして記述する
  - この章ではBODYタグに入るタグを重点的に学習する
- BODYの外のタグを誤って消さないように注意する

index.html

```
<!DOCTYPE HTML>      ……①
<html>               ……②
<head>               ……③
    <meta charset="utf-8">
    <meta name="viewport" content="width=device-width, initial-
    scale=1, maximum-scale=1, user-scalable=no">
    <meta http-equiv="Content-Security-Policy" content="default-src
    *; style-src * 'unsafe-inline'; script-src * 'unsafe-inline'
    'unsafe-eval'">                                              ……④
    <script src="components/loader.js"></script>      ……⑤
    <link rel="stylesheet" href="components/loader.css">
    <link rel="stylesheet" href="css/style.css">                 ……⑥
    <script>
        alert("こんにちは");                                      ……⑦
    </script>
</head>
<body>          ……⑧
    <br />      ……⑨
    はじめてのプログラム
</body>
</html>
```

第2章 HTML入門

<body>タグ内に記述するタグの種類

終了タグのあるタグ

タグ名	概要
h1	見出しを定義します。h1～h6まであり、h1が最も高レベル、h6が最も低レベルな見出しです。 例： 見出し
p	文章の段落を定義します。 例： 文章の段落を定義します。
div	特に意味を持たないタグです。複数のタグをまとめて扱うときや、四角い枠を描画したいときに使います。 例： 見出し 段落
a	リンクを定義します。 href属性・・・リンク先のURLを指定します。 例：TOPへ
button	ボタンを定義します。 例：<button>ボタン</button>

空要素（終了タグのないタグ）

タグ名	概要
img	画像を参照します。 src属性：画像の参照先を指定します。 alt属性：画像が何らかの理由で表示できなかった場合に、画像の変わりに表示する文字列を指定します。 例：<img src="flower.jpg" alt="花">

すべてのタグにつけられる属性

タグ名	概要
id	要素を識別するためのIDです。文書内で重複する値を指定することはできません。 例： …
class	CSSのクラス名を指定します。（→第3章） 例： …

第2章 HTML入門

リンク

リンクの設定

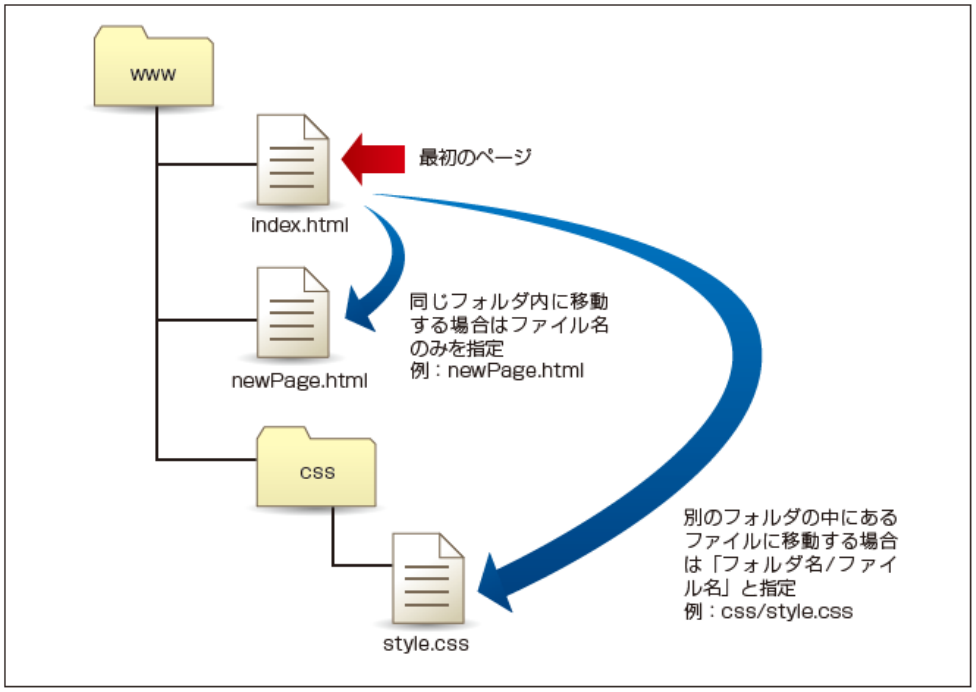
- Aタグを使用し、リンク先はパスをhref属性の値で指定
- 絶対パス指定
  - パスを全て記述する方法
- 相対パス指定
  - 現在のファイルからの相対的なパスを記述する方法

文法　リンクの設定

```
<a href="リンク先のパス">リンク文字列</a>
```

相対パス指定

- 対象ファイルまでの位置を相対的に指定する方法
  - 下位のフォルダを指定するときは「/」を使用
  - 上位のフォルダを指定するときは「..」と「/」を使用



第2章 HTML入門

実習

リンクの配置

```
<body>
  <a href="https://edu.monaca.io/">Monacaへ</a>
</body>
```

実行結果

※リンク先によってはプレビュー機能で動かない場合がある。
その場合はデバッガーアプリで確認。

準備

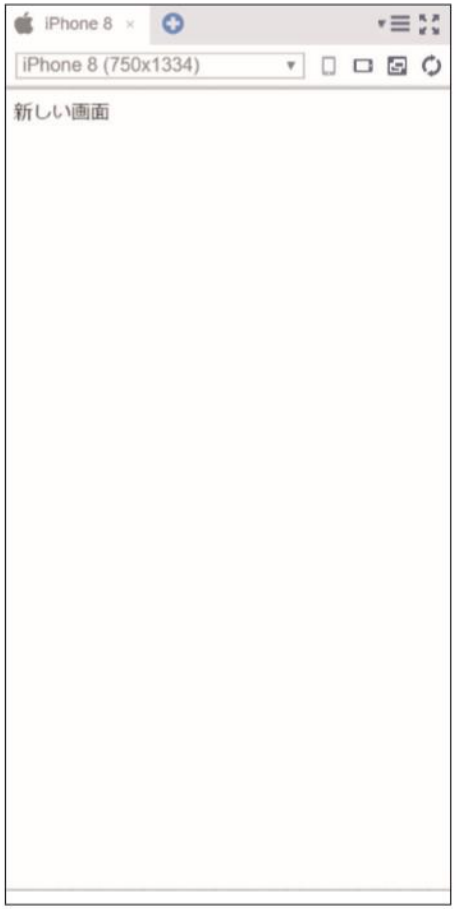
- 別ファイル「newPage.html」を確認

実習

- 相対パスのリンクを追記

```
<a href="newPage.html">次の画面へ</a>
```

実行結果



第2章 HTML入門

画像の表示

画像を表示する方法

- imgタグを記述
- src属性でパスを指定する

 文法　画像の表示

```
<img src="画像ファイルのパス">
```

- 例

```
<img src="monaca.jpg">
```

第2章 HTML入門

実習

準備

- 画像ファイル「monaca.jpg」を確認

実習

- imgタグを追記

```
<img src="monaca.jpg">
```

実行結果

第3章 CSS入門

第3章 CSS入門

CSSとは

CSS（Cascading Style Sheets）とは

- HTML文章を装飾するための技術
- 色やサイズなどを変更できる
- カスケーディングという特徴がある

CSSをHTMLファイルに読み込む方法

- 外部ファイル
  - linkタグで外部ファイルを指定して読み込む
- styleタグ
  - styleタグの中に記述する
- style属性
  - style属性の中に記述する

CSSをHTMLファイルに読み込む方法（外部ファイル）

- linkタグを記述
- href属性でパスを指定する

文法　CSSファイルの読み込み

```
<link rel="stylesheet" href="CSSファイルのパス">
```

- 例

```
<link rel="stylesheet" href="css/style.css">
```

CSSの書き方（1）

- セレクタ
  - 対象となる要素を指定する
- プロパティ
  - どのようなスタイルを適応するかを指定する

文法　セレクタとプロパティの記述方法

```
セレクタ {
        プロパティ: 値;
        プロパティ: 値;
        }
}
```

CSSの書き方（2）

- セレクタ
  - P(パラグラフ)タグを対象とする
- プロパティ
  - 文字の色を赤くする
  - 文字のサイズを10pxにする

文法　セレクタとプロパティの記述例

```
p {
        color: red;
        font-size: 10px;
}
```

第3章 CSS入門

セレクタの種類

タグセレクタ

- 対象要素をタグ名で指定

IDセレクタ

- 対象要素をID属性値で指定
  - セレクタの先頭に#をつける

クラスセレクタ

- 対象要素をクラス値で指定
  - セレクタの先頭に.(ドット)をつける

第3章 CSS入門

実習

実習

- 2章のプロジェクトを開く
- cssフォルダ内のstyle.cssを編集

```
a {
    font-size: 30px;
}
```

実行結果

実習

- index.htmlのaタグにID属性を付与

```
<a href="https://edu.monaca.io/" id="target">Monacaへ</a>
```

- style.cssにIDセレクタを利用したスタイルを記述

```
#target {
    color: red;
}
```

実行結果

実習

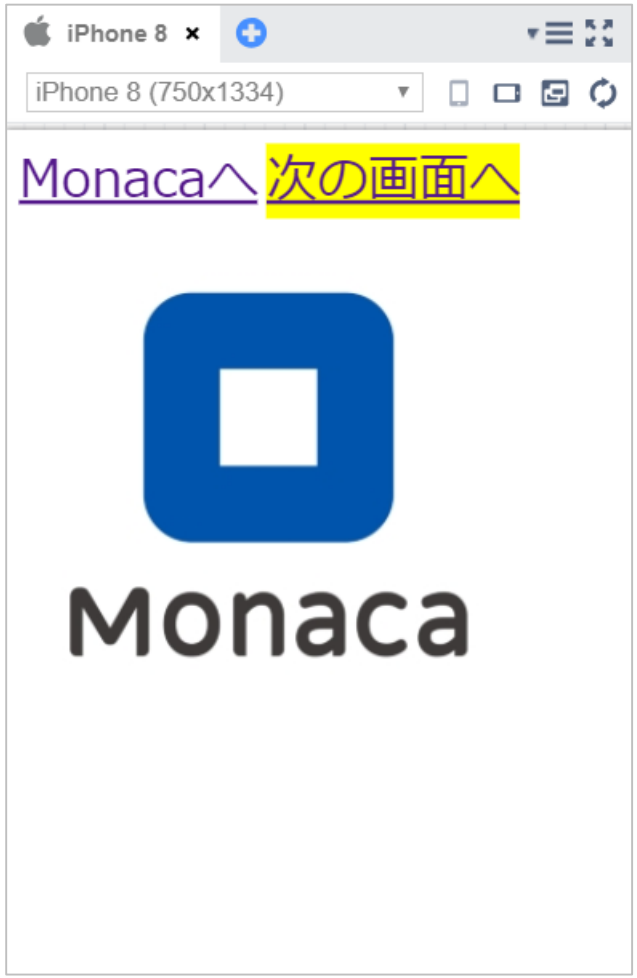
- index.htmlのaタグにクラス属性を付与

```
<a href="newPage.html" class="bright">次の画面へ</a>
```

- style.cssにクラスセレクタを利用したスタイルを追記

```
.bright {
    background-color: yellow;
}
```

実行結果



第3章 CSS入門

プロパティの種類

色を指定するプロパティ

プロパティ	説明	例
color	文字色を設定します。	color: red;
background-color	背景色を設定します。	background-color: red;
border	線の色（および線種と線 の太さ）を設定します。	border: solid 1px red; 線種、線の太さ、線の色の順に 設定します。solidは直線を表します。

カラーコード

- 光の三原色で色を作る方法
- カラーコードでは16進数を使う

カラーコードの例

- 紫の色を作る

```
#ff00ff
```

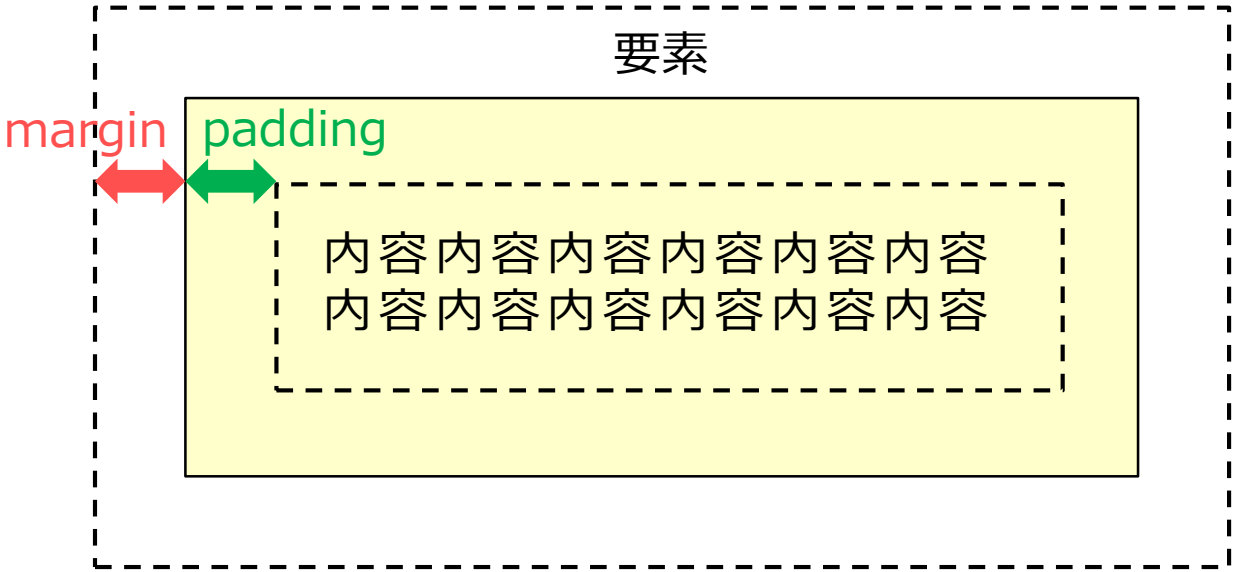
  - 赤がff（255）緑が00（0）青がff（255）
  - 原色の赤と青を混ぜた色＝紫になる

サイズや位置を指定するプロパティ

プロパティ	説明	例
font-size	文字のサイズを設定します。	font-size: 12px;
text-align	要素内の横方向の配置を設定します。	text-align: left;（左寄せ） text-align: right;（右寄せ） text-align: center;（中央揃え） text-align: justify;（均等割付）
width	要素の横幅を設定します。	width: 100px
height	要素の高さを設定します。	height: 300px;
margin	枠線の外側の余白を設定します。	margin: 20px;
padding	枠線の内側の余白を設定します。	padding: 10px;

marginとpadding

- どちらも余白の幅を指定するプロパティ
- marginはborder(枠線)の外側の余白
- paddingはborderの内側の余白



第3章 CSS入門

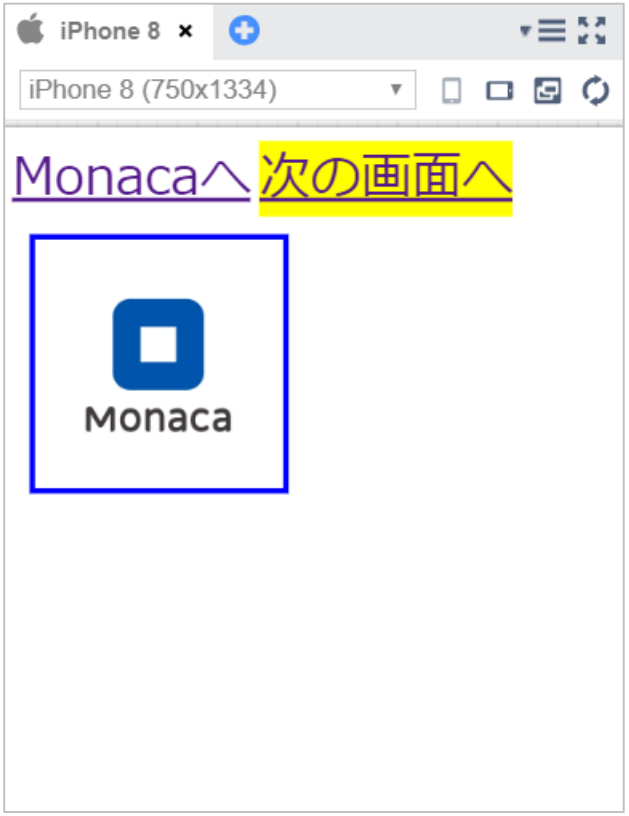
実習

実習

- style.cssを編集

```
img {
    width: 30%;
    border: solid 3px #0000ff;
    margin: 10px;
    padding: 20px;
}
```

実行結果



第4章 JavaScript入門

第4章 JavaScript入門

JavaScriptの書き方

scriptタグの中に記述する

```
<script>
  alert("こんにちは");
</script>
```

外部ファイルに記述する

```
<script src="JavaScript ファイルのパス"></script>
```

書き方のルール

- 基本的に半角の英数字と記号のみを使用する
  - 「'」か「"」で括れば全角文字も利用可能
- 大文字と小文字は別の文字として扱われる
- 命令文の末尾には「;」をつける。
- ひとまとまりの命令群を波かっこ{ } で囲む
  - 囲まれた範囲をブロックと呼ぶ
- 改行や半角スペースは自由に挿入できる

改行やスペースの活用例

例　プログラムA(改行とスペースを使わない)

```
for(i=0;i<10;i++){alert(i);}
```

例　プログラムB（改行とスペースを使う）

```
for(i = 0; i < 10; i++) {
    alert(i);
}
```

インデントとは

- 見やすいように字下げを行うこと
  - 空白スペースやタブなどを用いる
  - 波かっこの対応関係が一目でわかるようになる

コメントとは

- メモを書いたり命令を無効化したりできる

文法　一行コメント

```
//alert("こんにちは");
```

文法　複数行コメント

```
/*
コメントとして記述した内容は、
スクリプトには影響しません。
*/
```

第4章 JavaScript入門

データの扱い方

変数の作り方

- メモリ上に変数を作る作業を変数の「宣言」と言う
- 変数は名前を付けて管理できる

文法　変数宣言の書式

```
var 変数名;
```

例　xという名前の変数を作る

```
var x;
```

変数の使い方

- 変数を作った直後は空の状態
- 変数にデータを入れるには代入を行う
- 宣言と代入は1 行にまとめて同時に行うこともできる

文法　変数へ値を入れる

```
変数名 = 値;
```

文法　宣言と代入を同時に行う

```
var 変数名 = 値;
```

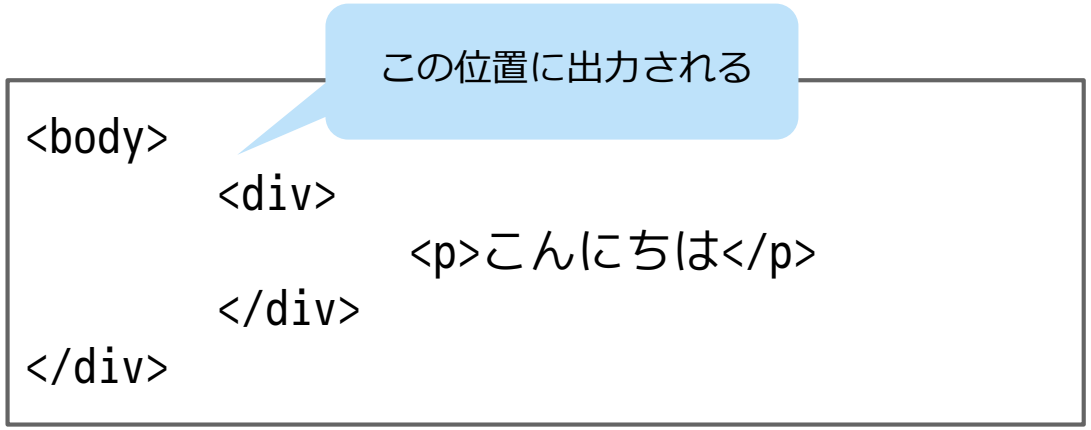
JavaScript から画面にデータを出力する

- document.writeln()で簡単に値を画面に出力できる
  - ※簡易的ですが実務は推奨されない命令です

文法　変数へ値を入れる

```
document.writeln(表示するデータ);
```

結果位置



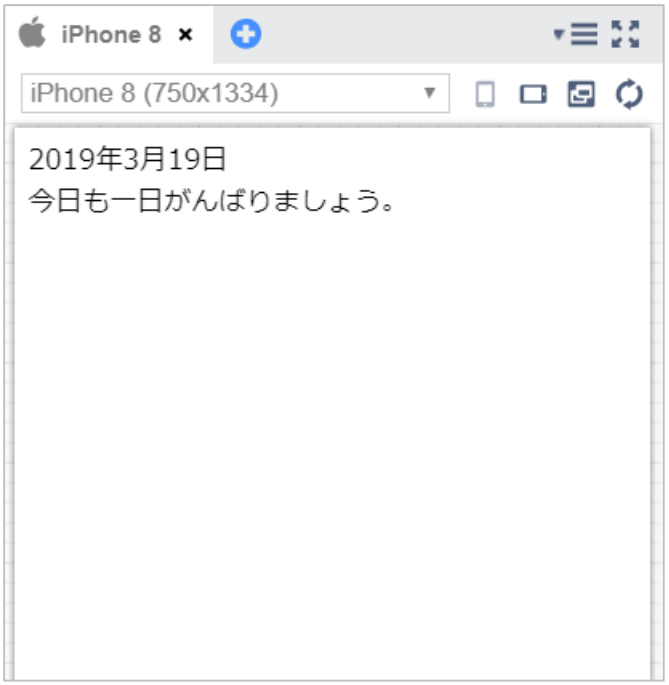
第4章 JavaScript入門

実習

実習

- index.htmlのbodyタグ内を編集
- index.htmlのscriptタグ内を編集

結果



今日の日付をプログラムで取得する方法

- 日付オブジェクトの準備

```
var 変数 = new Date();
```

- 年を取得する命令

```
変数.getFullYear();
```

- 月を取得する命令

```
変数.getMonth();
```

- 日を取得する命令

```
変数.getDate();
```

実習

- index.htmlのscriptタグ内を編集

```
<script>
  // 日付に関する命令を使えるようにする
  var date = new Date();
  // 年、月、日の取得
  var year = date.getFullYear();
  var month = date.getMonth() + 1;
  var day = date.getDate();
  // 日本の表記にする
  var today = year + " 年" + month + " 月" + day + " 日";
  document.writeln(today);
</script>
```

第5章 条件分岐

第5章 条件分岐

if文

条件分岐とは

- 条件に応じて処理の流れを分岐する仕組み
- if文は正しいか否かの2値(真偽値)を使って分岐する
- 条件は条件式で記述できる

if文の書き方

- 条件と正しい場合に実行する処理を記述
- elseを使うことで正しくない場合に実行する処理も記述できる
  - elseは省略が可能

文法　if文の書き方

```
if (条件式) {
    条件式が正しい場合に実行する処理
} else {
    条件式が正しくない場合に実行する処理
}
```

条件式

- 比較演算式は「文字列」や「数値」を比較して真偽値を返す

演算子	概要	条件式の例	結果
==	左辺と右辺が等しい場合は正しい	1 == 1	正しい
!=	左辺と右辺が等しくない場合は正しい	1 != 2	正しい
<	左辺が右辺より小さい場合は正しい	1 < 1	正しくない
<=	左辺が右辺以下の場合は正しい	1 <= 1	正しい
>	左辺が右辺より大きい場合は正しい	1 > 1	正しくない
>=	左辺が右辺以上の場合は正しい	1 >= 1	正しい

第5章 条件分岐

実習

準備

- 第４章で作成したプロジェクトを開く

実習

- scriptタグにif文を使ったプログラムを追記

実行結果

第5章 条件分岐

多方向分岐

多方向分岐とは

- 条件に応じて処理の流れを多方向に分岐する仕組み
- 例えば、曜日毎にメッセージを変えたい場合に利用できる
  - if文を7個書けば可能だが冗長である
- else if文を使えば分かりやすく記述できる

else if文の書き方

- else if 文は何回でも記述できる

文法　if文の書き方

```
if (条件式1) {
    条件式1 が正しい場合に実行する処理
} else if(条件式2) {
    条件式2 が正しい場合に実行する処理
} else {
    条件式が正しくない場合に実行する処理
}
```

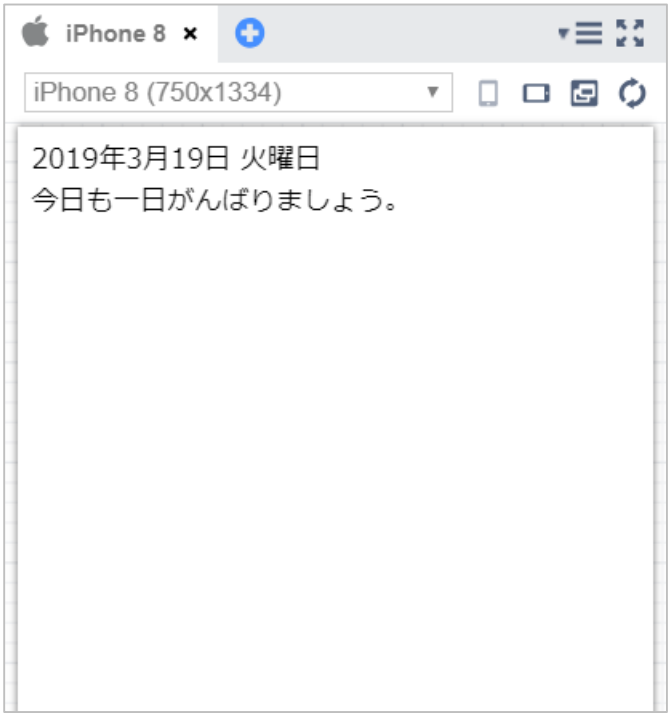
第5章 条件分岐

実習

準備

- 先ほどのプログラム開いて修正する

実行結果



第6章 関数

第6章 関数

関数

関数とは

- 処理をひとまとめにしたもの
- 『引数』を受け取る
- 処理結果を『戻り値』として返す
- 『引数』と『戻り値』は省略可能

関数の書き方

- まずは引数・戻り値ともに省略したパターンを確認

文法　引数・戻り値ともに省略したパターン

```
function 関数名() {
  処理;
}
```

文法　関数の呼び出し

```
関数名();
```

第6章 関数

実習

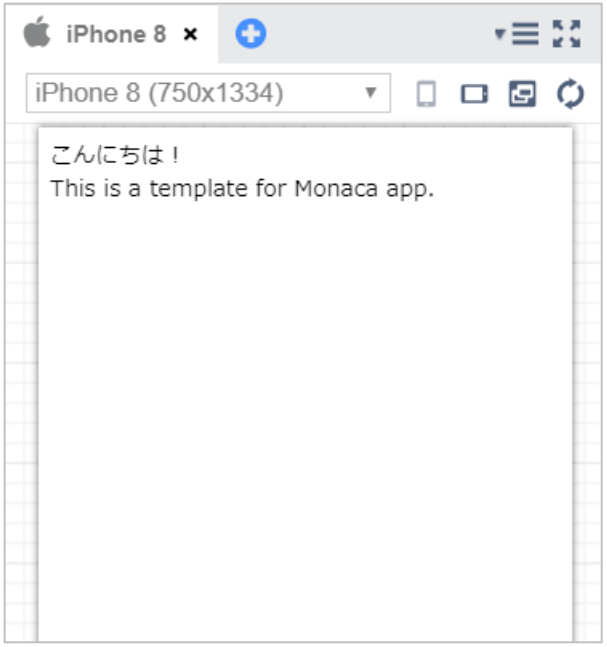
準備

- プロジェクトを新規作成

実習

- scriptタグに関数を使ったプログラムを記述

実行結果



第6章 関数

引数がある関数

引数がある関数の書き方

- 関数には引数を渡すことができる
- 引数はカンマ区切りで複数指定することも可能

文法　関数の定義（引数あり）

```
function 関数名(引数を入れる変数名) {
  処理;
}
```

文法　関数の呼び出し

```
関数名(関数に渡す引数);
```

第6章 関数

実習

準備

- 先ほどのプロジェクトを変更

実習

- scriptタグに関数を使ったプログラムを記述

実行結果

第6章 関数

戻り値がある関数

戻り値がある関数の書き方

- 関数は処理結果を戻り値として返せる
- 戻り値は一つしか返せない

文法　関数の定義（戻り値あり）

```
function 関数名() {
  処理;
  return 戻り値;
}
```

文法　関数の呼び出し

```
var 戻り値を入れる変数名 = 関数名();
```

第6章 関数

実習

準備

- 先ほどのプロジェクトを変更

実習

- 19行目以降にプログラムを追記

実行結果

準備

- 先ほどのプロジェクトを変更

実習

- 30行目以降にプログラムを追記

実行結果

第7章 イベント

第7章 イベント

イベント

イベントとは

- 起こった出来事のことを「イベント」と呼ぶ
  - ボタンのクリック/タッチ、画面スワイプなど
- イベント発生時に特定の関数を呼び出すことができる
  - 例：ボタンを押したらalert()関数を呼びだすなど

文法　HTML 要素にイベントを関連付ける

```
<タグ名 on イベント名="関数名()">
```

例　ボタンがクリック時にtest関数を実行する

```
<button onclick="test()">ボタン</button>
```

第7章 イベント

実習

準備

- 『7章イベントの練習』をインポート

実習

- index.html を編集（関数とonclickを記述）

実行結果

- ボタンクリック時にダイアログが表示される

準備

- 先ほどのプロジェクトを変更

実習

- index.html を編集（関数とonloadを記述）

実行結果

- プレビュー確認時にダイアログが表示される

第8章 DOM

第8章 DOM

DOM

DOMとは

- DOM（Document Object Model）
- JavaScriptでHTMLの各要素にアクセスする仕組み

DOM が利用可能になるタイミング

- ブラウザがHTMLを読み込んだ後
- JavaScriptの書き方に工夫が必要
- loadイベントなどを活用する

HTML要素へのアクセス

- document.getElementById()命令を利用

文法　要素へのアクセス

```
document.getElementById("ID 値")
```

要素の内容変更

- innerHTMLにより内容を変更できる

文法　内容の変更

```
document.getElementById("ID 値").innerHTML = "書き換える内容";
```

第8章 DOM

実習

準備

- 『8章あいさつアプリ』をインポート

実習

- index.htmlとstyle.cssを編集
- greet()関数を定義
- onloadでgreet()関数を呼び出す

実行結果

- 時間帯に応じたあいさつが表示される

第8章 DOM

要素の属性変更

要素の属性変更

- innerHTMLのかわりに属性名を指定

文法　属性の変更

```
document.getElementById("ID 値").属性名 = "属性値";
```

例　画像を「flower.jpg」に切り替える

```
document.getElementById("target").src = "flower.jpg";
```

スタイル属性変更

- style属性のプロパティを指定

文法　スタイルの変更

```
document.getElementById("ID 値").style.CSS プロパティ名 = "値";
```

スタイル属性変更

- style属性のプロパティを指定

例　要素の外側余白を20px に設定

```
document.getElementById("target").style.margin = "20px";
```

例　要素の背景色を青に設定する

```
document.getElementById("target").style.backgroundColor = "blue";
```

第8章 DOM

実習

準備

- 先ほどのプロジェクトを変更

実習

- greet()関数を修正

実行結果

- 時間帯に応じた画像が表示される

第9章 フォーム

第9章 フォーム

フォーム

フォームとは

- ユーザーに情報を入力させるための部品
  - テキストボックス、チェックボックス、プルダウンメニューなど
- HTMLタグで作成できる
- JavaScriptでフォームの値を取得できる

文法　テキストボックス

```
<input type="text">
```

例　テキストボックスの記述

```
<input type="text" value="テスト">
```

テスト

文法　チェックボックス

```
<input type="checkbox">
```

例　チェックボックスの記述

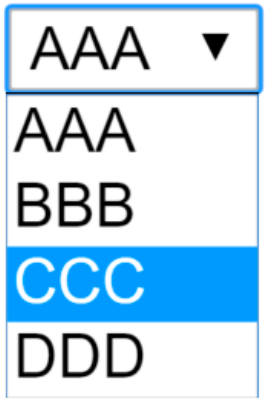
```
<input type="checkbox" checked>
```

文法　ドロップダウンメニュー

```
<select>
<option value="選択肢の値">表示する選択肢</option>
<option value="選択肢の値">表示する選択肢</option>
…
</select>
```

例　ドロップダウンメニューの記述

```
<select>
<option value="1">AAA</option>
<option value="2">BBB</option>
<option value="3">CCC</option>
<option value="4">DDD</option>
</select>
```



第9章 フォーム

実習

準備

- 『9章フォームの練習』をインポート

実習

- index.html を編集（関数を記述）

実行結果

- フォームに値を入力後、完了ボタンを押す
- alertに入力内容が表示されれば成功

第10章 いろいろな演算子

第10章 いろいろな演算子

いろいろな演算子

演算子とは

- 足し算を行う「+」や、比較を行う「==」などのこと
- 演算子には幾つかの種類がある
  - 四則演算子
  - 論理演算子
  - 複合代入演算子

四則演算子

演算子	概要	条件式の例	結果
+	数値の加算	1 + 1	2
+	文字列の結合	"Hello" + "World"	"HelloWorld"
-	数値の減産	2 - 1	1
*	数値の乗算	2 * 2	4
/	数値の除算	10 / 2	5

論理演算子

演算子	概要	条件式の例	結果
&&	2つの条件式がどちらも trueの場合はtrue	1 == 1 && 2 == 2	true
||	2つの条件式のどちらかがtrueの場合はtrue	1 == 2 || 2 == 2	true

複合代入演算子

演算子	概要	使用例	結果（a の値）
+=	左辺の値に右辺の値を加算したものを代入	a = 1; a += 2;	3
-=	左辺の値から右辺の値を減算したものを代入	a = 5; a -= 2;	3
*=	左辺の値に右辺の値を乗算したものを代入	a = 3; a *= 2;	6
/=	左辺の値を右辺の値で除算したものを代入	a = 10; a /= 2;	5
++	変数に1加算する（インクリメント）	a = 1; a++;	2
--	変数から1減算する（デクリメント）	a = 1; a--;	0

BMI の求め方

- 四則演算子と論理演算子でBMI 計算を行う
- BMIは身長と体重から求める体格指数
- 痩せ・標準体重・肥満などの判定が行える

BMIの計算式

```
体重 ÷（身長2）
```

※体重の単位はkg、身長の単位はm（cmではないことに注意）

例　身長170ｃｍ、体重60kgの人の場合

```
60 ÷ (1.72) = 20.761...
```

第10章 いろいろな演算子

実習

準備

- 『10章BMI計算アプリ』をインポート

実習

- index.html を編集（関数を記述）

実行結果

- フォームに値を入力後、計算ボタンを押す
- 画面にBMIが表示されれば成功

第11章 配列

第11章 配列

配列

配列とは

- 複数のデータを一括で管理するための仕組み
  - 変数という箱に仕切りで部屋を作るイメージ
  - この部屋のことを「要素」と呼ぶ
  - 要素には文字や数字、また配列などを格納できる

文法　配列の作成

```
var 配列名 = [要素1, 要素2, 要素3, …];
```

例　配列の記述例

```
var fruits = ["りんご", "みかん", "いちご", "ぶどう"];
```

文法　配列内の要素を参照する

```
配列名[インデックス]
```

例　配列内の要素を参照する記述例

```
var fruits = ["りんご", "みかん", "いちご", "ぶどう"];
alert(fruits[2]);
```

文法　配列の要素数を参照する

```
配列名.length
```

例　配列の要素数を取得する記述例

```
var fruits = ["りんご", "みかん", "いちご", "ぶどう"];
alert(fruits.length);
```

第11章 配列

実習

準備

- 『11章心理テスト』をインポート

実習

- index.html を編集（scriptタグ内を記述）

実行結果

- 心理テストを回答
- 画面に結果が表示されれば成功

第12章 繰り返し

第12章 繰り返し

繰り返し

繰り返しとは

- 同じ処理を何回も繰り返し実行したい場面がある
  - 例
    - 1～100 までの数の合計を求める
    - 100個分の要素を持つ配列のデータを 1 件ずつ表示
- コピー＆ペーストでの対応は望ましくない
- 繰り返しのための文法で対応

for文

- 中括弧の中の処理を繰り返す
- 繰り返しの諸条件をfor()の括弧内にまとめて記述できる

文法　for文

```
for（初期化式; 継続条件式; 増減式）{
  繰り返す処理
}
```

式の種類	概要	実行するタイミング
初期化式	カウンタ変数を初期化する代入式	最初の１回だけ実行される
継続条件式	繰り返しを継続する条件式	１回分の繰り返し処理の最初に毎回実行される
増減式	カウンタ変数の値を増減させる計算式	１回分の繰り返し処理の最後に毎回実行される

第12章 繰り返し

実習

準備

- 『12章繰り返しの練習』をインポート

実習

- index.html を編集（scriptタグ内を記述）

実行結果

- １～１０までの数字が表示される

準備

- 先ほどのプロジェクトを変更

実習

- index.html を編集（scriptタグ内にinsert()関数を追記）

実行結果

- 画像が5つ表示される