

# アプリづくりの準備



ここからは「Monaca (モナカ)」という、スマホアプリをつくるためのWebサイトを使うよ。Monacaを使えば、特別なアプリをインストールしなくてもWebブラウザだけでアプリを開発することができるんだ。つくっているアプリを試しに動かしたりも簡単にできて便利だよ。Monacaへアクセスするときには、Part 4でも使ったChromeを使うようにしよう。ほかのWebブラウザだとうまく動作しないことがあるから注意してね。Macを使っている場合は、Mac用のChromeでアクセスすれば使い方はいっしょだよ。



## Monacaのアカウント登録

Part 5の内容を行うには、Monacaのサイトで無料登録をして、アカウントを作成する必要があります。プログラミングサポートページの「Monacaのアカウント登録」の項目で紹介している手順で、大人といっしょに必要な手続きを行ってください。

<https://kodomonokagaku.com/miraiscience/support/>



## 1 ダッシュボードを開く

Monacaで作業をはじめるためには、Dashboard (ダッシュボード) というページを使うよ①。このダッシュボードのページは、つくっているアプリの一覧を見ることができ、選択するとアプリの内容を編集することができる。Scratchでいうところの、「私の作品」のページと同じだね。



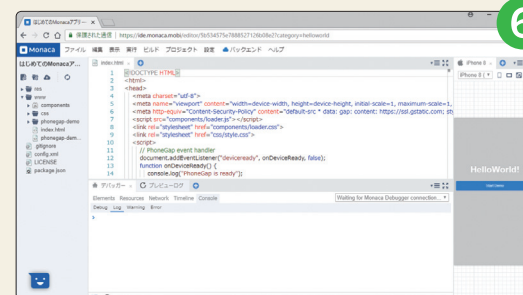
## 2 Monacaにログイン

ダッシュボードのページを見るためには、Monacaのサイトにログインしておく必要がある。一度作業をやめて、Monacaのサイトにログインしていないときは、ログインしてから作業をはじめよう。画面右上にあるログインを選択して②、メールアドレスとパスワードを入力して、ログインのボタンをクリックしよう③。これもScratchを使う場合といっしょだから大丈夫だよ。

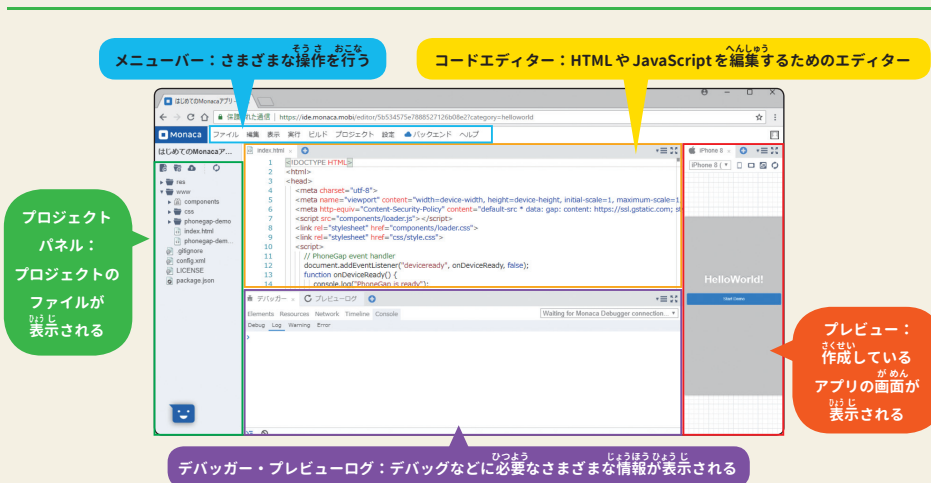


## 3 クラウド IDE

Monacaの無料 (Free) プランを使う場合は、アプリのプロジェクト (Scratchのプロジェクトと同じだ) は3つまでしかつくれないという制限がある。アカウント作成をした段階で、はじめから「はじめての Monaca アプリ」というサンプルのプロジェクトが用意されているから、まずはそれを開こう。プロジェクトの一覧からプロジェクトを選択して④、「クラウド IDEで開く」というボタンをクリックしよう⑤。



クラウド IDE  
の使い方は  
次のページへ！



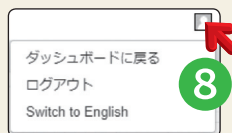
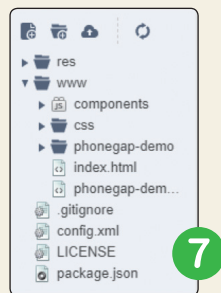
IDEというのは、統合開発環境 (Integrated Development Environment) の略で、プログラムを書くためのエディターだけでなく、バグの発見に役立つデバッガーというツールや、つくったアプリをスマホに転送するためのツールなど、プログラミングをするときに必要になるいろいろなツールをまとめて、使いやすくしたものことだよ。

画面の真ん中に表示されているのがコードエディターだ。ここにHTMLやJavaScriptなどを入力していくよ。

画面の左に表示されているのは、プロジェクトパネルで、ここにアプリで使っているさまざまなファイルが表示されている。この一覧からHTMLやJavaScriptを書いたファイルをダブルクリックすると、コードエディターで開いて編集ができる。

画面の右に表示されているのはプレビューだ。つくっているアプリの見た目がどのようになっているかを表示している。Part 4の145ページで紹介したBracketsのライブプレビューと同じだと思えばいい。画面の下にあるのはデバッガー・プレビューログだ。プログラムのエラーや、アプリの動作の情報などを確認することができるよ。

画面の右上には、人型のアイコンが表示されているね。ここをクリックすると、ダッシュボードの画面に戻ったり、Monacaのサイトからログアウトしたりすることができるぞ。



# あいさつアプリ をつくろう



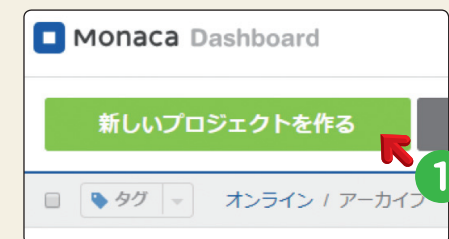
最初はシンプルなアプリをつくりながら、Monacaの使い方や基本的なアプリのつくり方を解説するよ。最初につくるのは「あいさつを表示してくれるアプリ」だ。

## 新しいプロジェクト

さっそく新しくプロジェクトをつくってみよう。MonacaのプロジェクトはScratchのプロジェクトと同じで、これからつくるアプリを管理するための入れ物のことだよ。

ダッシュボードの画面の左上にある「新しいプロジェクトを作る」というボタンをクリックしよう。プロジェクトのテンプレートを選択する画面が表示される。テンプレートは、アプリをつくるために必要なファイルの「ひな形」のことだよ。つくりたいアプリに応じて、いくつかの種類が用意されているんだ。今回はシンプルなアプリをつくって練習をしたいから、一番右の「No Framework」というボタンを押す。

「最小限のテンプレート」で「作成」を選択しよう。プロジェクト名を入力する画面が表示されるので、「あいさつアプリ」と入力して、「プロジェクトを作成する」を選択しよう。今回は説明については入力しなくてOKだ。



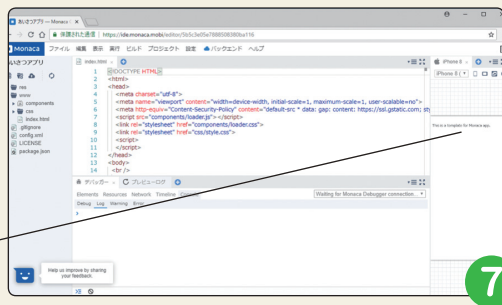
## 2 IDEを開く

プロジェクトの一覧に「あいさつアプリ」が追加されたね。「クラウド IDE で開く」を選択して IDE の画面に切り替えよう⑥。「あいさつアプリ」を編集できる IDE の画面が開いたね⑦。

プレビューには、「This is a template for Monaca app. (これは Monaca でつくるアプリのひな形です)」と表示されているはずだ。

プレビューの  
さいしょの表示

This is a template for Monaca app.



## 3 HTMLの編集

コードエディターで index.html が開いていることを確認して、次の文字列が書かれている部分を探してみよう。

14 行目に改行のための「<br/>」のタグがあり、その下の 15 行目に「This is a template for Monaca app.」という文字列があるね。これを「あいさつアプリ」という一番大きな表題に書き換えてみよう⑧。表題を書くためのタグは覚えているかな？「h1」のタグだったね。Monaca のコードエディターも、Brackets と同じように、タグの候補を自動的に表示してくれる機能 (141 ページ参照) があるから活用しよう。

8

14 行目と 15 行目を削除

あいさつアプリという表題を  
表示するように変更

```
13 <body>
14 <br />
15 This is a template for Monaca app.
16 </body>
```

```
13 <body>
14 <h1>あいさつアプリ</h1>
15 </body>
```

## 4 保存する

うまく修正できたら、ファイルメニューから、「保存」を選択⑨して、編集した index.html を保存しよう。

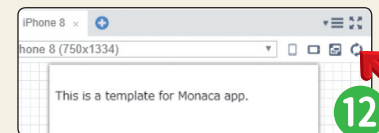
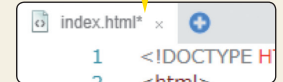
編集をして、保存をしていないファイルについては、エディターの上にあるファイル名の最後に「\*」のマークが表示されるよ⑩。保存するとこのマークは消えるぞ。

保存をすると自動的にプレビューが更新されて、表示されている文字が「あいさつアプリ」に変わったはずだ⑪。ここまでは Part 4 でやった HTML の編集とほとんど同じだから簡単だよ。

もし、プレビューが自動的に更新されない場合、更新のためのボタン (丸い矢印) を押せば⑫、最新のコンテンツに更新されるよ。



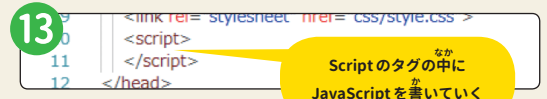
保存をしていない印



## 5 JavaScriptとは？

次は JavaScript を書いてみよう。

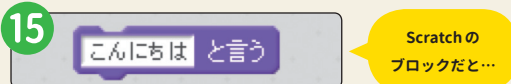
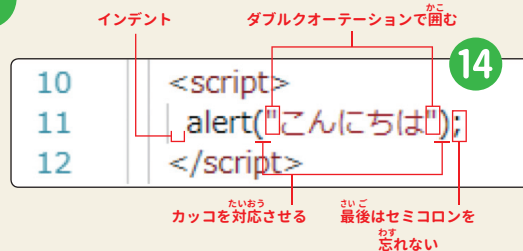
JavaScript は Web ブラウザーで実行することができるプログラミング言語だ。JavaScript を使えば、いろいろなしくみをもった Web サイトをつくることができる。今回は Web サイトではなく、スマホアプリをつくっているけれど、スマホの OS には、HTML を表示したり、JavaScript を実行したりするための部品 (WebView と呼ばれる) が内蔵されている。HTML で画面をつくり、JavaScript を使ってしくみをつくれば、スマホのアプリとして動作させることができるんだ。HTML ファイルの中に「script」というタグを書いて、その中に JavaScript を書くことができるぞ⑬。





## 6 JavaScript を書こう

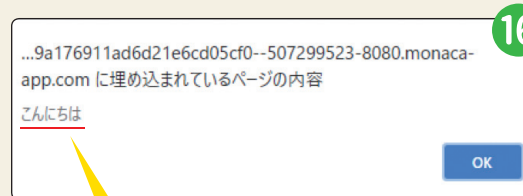
index.html の 10 行目と 11 行目を見てみよう。じつは「script」タグはすでに用意されているぞ。この script タグの中に JavaScript を書けば、index.html が表示されたときに実行されるよ。



あいさつアプリという名前にしたけれど、まだ何のあいさつもしてくれないよね。「アラート」という小さなウィンドウを表示させて、そこにあいさつを表示させるようにしてみよう。アラートを表示させて、その中にメッセージを表示させるための JavaScript は 14 のようになるぞ。「こんにちは」以外はすべて半角で入力することに注意しよう。Scratch の「&quot;と言う&quot;のブロック 15 と同じだと考えるとわかりやすいね。Brackets と同じように、自動的にインデント (141 ページ参照) がつくので、script の開始タグ (10 行目) の最後で改行を入力すれば、インデントされた状態で次の行にカーソルが移動するぞ。

## 7 保存する

入力できたら、保存をしよう。保存をすると、16 のようなアラートが表示される。もし表示されない場合は、JavaScript が間違っていないかを確認してみよう。少しわかりにくいけれど、これは Monaca のプレビューで、今書いたアラートが表示されたということだよ。



指定したあいさつはこの部分に表示されている

Scratch だと...



## 8 スマホで動作確認しよう

つくったあいさつアプリをスマホで動かしてみよう！ Monaca には「実機デバッグ」という機能があって、つくったアプリをスマホで試しに動かしてみることができるよ。



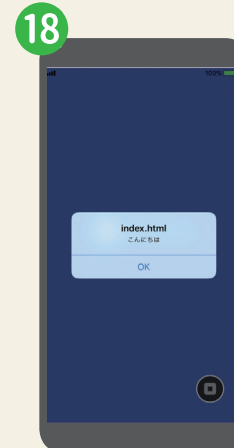
### Monaca デバッガーのインストール

つくったスマホアプリを実際のスマホで動かすためには、7 ページにある環境を備えたスマホの機種を用意した上で、「Monaca デバッガー」という無料のアプリをインストールする必要があります (iPhone、Android 対応)。以下のプログラミングサポートページの「Monaca デバッガーのインストール」の項目で紹介している手順でアプリをインストールして、Monaca と同じメールアドレスとパスワードでアプリにログインしてください。

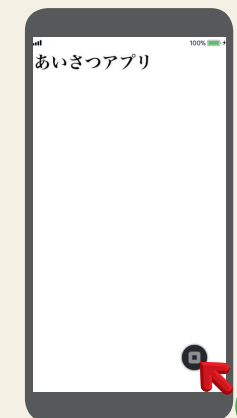
<https://kodomonokagaku.com/miraiscience/support/>



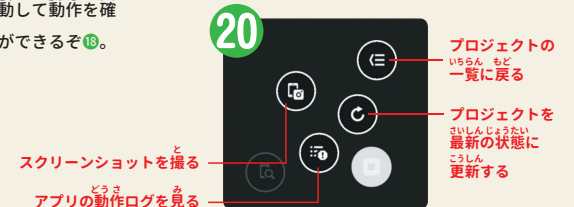
Monaca デバッガーのアプリにログインすると、Monaca.io プロジェクトの欄に今つくっている「あいさつアプリ」が見えるはず。これをタップする 17。



アプリを起動して動作を確認することができるぞ 18。



画面の右下に表示されているのは、デバッガーメニューを表示するためのボタンだよ 19。これを押すと、メニューが表示されて、いろいろな操作ができるぞ 20。





# あいさつアプリを 改造しよう



ここまでつくったあいさつアプリは、アプリを起動するとすぐにアラートが表示されていたね。「イベント」と「関数」を使って、アプリを起動した後で、ボタンを押すとあいさつを表示するように改造してみよう。まずは下に、次の①から③で作成・編集するHTMLやJavaScriptを整理しておくよ。どこを書いているかわからなくなったら、このページに戻って確認しよう。

```
1 <!DOCTYPE HTML>
2 <html>
3 <head>
4   <meta charset="utf-8">
5   <meta name="viewport" content="width=device-width, initial-scale=1, maximum-scale=1,
6   <meta http-equiv="Content-Security-Policy" content="default-src * data: gap: content:
7   <script src="components/loader.js"></script>
8   <link rel="stylesheet" href="components/loader.css">
9   <link rel="stylesheet" href="css/style.css">
10  <script>
11    function hello(){
12      alert("こんにちは");
13    }
14  </script>
15 </head>
16 <body>
17   <h1>あいさつアプリ</h1>
18   <button onclick="hello();">あいさつをする</button>
19 </body>
20 </html>
```



## ② 関数をつくる



## ① ボタンの設置



## ③ 関数の呼び出し



## ① ボタンの設置

「イベント」というのは、ユーザーがボタンをクリックしたり、ページを移動したり、ユーザーの操作によって生じる「できごと」のことだよ。ここでは、ボタンを押したらあいさつを表示するようにアプリを改造するぞ。

まずはボタンを表示させるためのHTMLをindex.htmlに加えよう。ボタンを表示させるためには、「button」というタグを使うよ。お手本を参考にして追加してみよう①。buttonというタグを使うと、HTMLでクリックできるボタンを表示させることができる。タグで囲んだ内容はボタンのラベルになるよ。今回は「あいさつをする」というラベルにしてみた。HTMLが完成したら、保存をして、プレビューでボタンが表示されたか確認してみよう②。

スマホでMonacaデバッガーを表示させたままにしておくと、更新を自動で検知して、表示を更新してくれるぞ③。

```
14 <body>
15   <h1>あいさつアプリ</h1>
16   <button>あいさつをする</button>
17 </body>
```

1

## あいさつアプリ

あいさつをする

2

3

## あいさつアプリ

あいさつをする

ボタンが  
ついたよ！



## 2 関数をつくる(定義する)

さて、今のままではボタンを押してもなにもおこらないし、index.htmlを保存す

るとアラートが表示されてしまうね。これを改善してみよう。

まずは、JavaScriptのアラートを表示している部分を「関数」として書き直す必要がある。関数というのは、プログラムの処理をまとめて、名前をつけたものだ。今回は、アラートを表示する処理をまとめて、helloという名前をつけた関数をつくってみよう

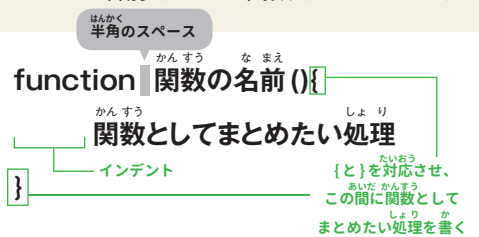
### 4. 関数の作り方の基本は右の図の

通りだ。関数をつくることを、関数を「定義する」ともいうよ。関数の名前は半角英数でつけることに注意しよう。

```
10
11
12 <script>
13   function hello(){
14     alert("こんにちは");
15   }
16 </script>
```

カッコを忘れないように

4



## 3 関数の呼び出し

```
16 <body>
17 <h1>あいさつアプリ</h1>
18 <button onclick="hello()">あいさつをする</button>
19 </body>
```

ダブルクォーテーションで囲む

5

helloという関数の呼び出し

helloという名前の関数がつくれたら、ファイルを保存しよう。ファイルを保存してもアラートは表示されなくなったね。関数をつくっただけでは、関数としてまとめた処理は実行されないんだ。関数の名前を指定して「呼び出し」をすると、関数としてまとめた処理が実行されるようになるぞ。

今回は、ボタンが押されたらこのhelloという関数を呼び出すようにしてみる。5。ボタンのHTMLにクリックされたイベントを調べて、helloという関数を呼び出すようなしくみを追加しよう。ここでは、ボタンのHTMLにonclickという部分を追加して、helloという名前の関数を呼び出すようにしているよ。

関数呼び出しの書き方

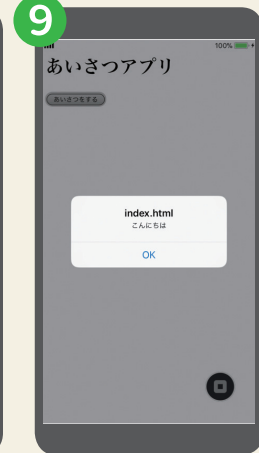
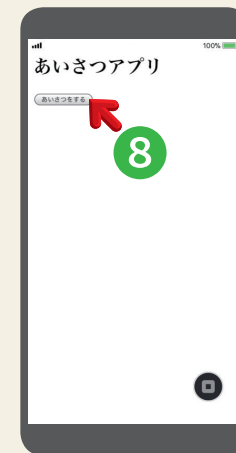
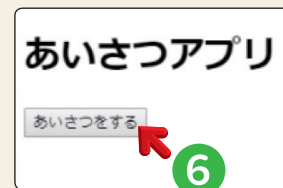
関数の名前();

さいご最後はセミコロンを忘れない

## 4 クリックして確認

保存をしたら、プレビューに表示されているボタンをクリックして6、アラートが表示されることを確認しよう7。

Monacaデバッガーでも同じように動作を確認してみよう89。



じつはこの関数に似たものをつくる機能はScratchにも用意されている

よ。Scratchでは「その他」のカテゴリから、新しいブロックを自分でつくることができる。たとえば、helloというブロックをつくって、内容を「こんにちは」という吹き出しが表示されるようにしておく。スプライトがクリックされたら、helloというブロックを実行するようにしたのと同じことだよ。少し違うのは、今回つくったHTMLでは、ボタンという別の部品(スプライト)がクリックされたらhelloという関数(ブロック)を実行できるようにしたことだね。

Scratchのブロックだと...



JavaScriptだと...

```
function hello(){
  alert("こんにちは");
}
```

```
16 <body>
17 <h1>あいさつアプリ</h1>
18 <button onclick="hello()">あいさつをする</button>
19 </body>
```

## 5 入力の受け取り

```
10 <script>
11 function hello(){
12   var name = prompt("お名前は?");
13   alert(name + "さん、こんにちは");
14 }
15 </script>
```

10

最後の仕上げに、162ページのHTMLとJavaScriptからさらに、自分の名前を入力して、あいさつに含めるようにしてみよう。⑩のように書いていくよ。完成した動作は⑪のようになる。Monacaデバッガーでも、名前を入力してあいさつを表示させることができるぞ。



11

では、書いたプログラムのポイントを解説していこう。ユーザーからの入力を受け取るときは、「prompt」という命令を使うよ。そして、JavaScriptでもScratchと同じように変数を使うことができるぞ。

12行目は入力された名前を保存しておく部分だね。ここでは名前を保存しておくためにnameという変数を用意した。JavaScriptの場合は、

はんかく 半角のスペース      へんすう 変数 name にゆうりよく 入力された文字列を代入

```
var name = prompt("お名前は?");
```

いりよう 入力用のボックス (プロンプト) を表示

さいご 最後はセミコロンを わす 忘れない

へんすう もじ 変数と文字をつなげる

```
alert(name + "さん、こんにちは");
```

へんすう 変数 name の内容と 「さん、こんにちは」をつなげて表示する

あと はんかく い var の後に半角スペースを入れて、変数の名前を書くよ。Scratchの場合は、「変数を作る」というボタンを押して、変数の名前を入力したよね。JavaScriptの場合は変数の準備についても文字で書くぞ。

Scratchと違って、変数の名前に漢字やひらがなは使えないから注意しよう。変数に代入する場合

は= (イコール) を使う。この場合は、ユーザーが入力用のボックス (プロンプト) に入力した文字列 (名前) を代入している。Scratchの場合は「=」の記号は代入ではなく、比較をするときに使っていたから注意しよう。

13行目では、変数nameには、入力された名前が代入されているから、これと「さん、こんにちは」という文字列をつなげて表示している。変数と文字列をつなげるために+ (プラス) の記号を使うよ。

Scratchにも入力用のボックスを表示するための、「と聞いて待つ」というブロックが用意されているから、ほとんど同じことができる。参考のためにScratchで同じことをした場合のスクリーンショットも載せておくね⑫。Scratchの場合は、入力された文字列は「答え」という変数に代入されるようになっているのがJavaScriptと違うところだね。

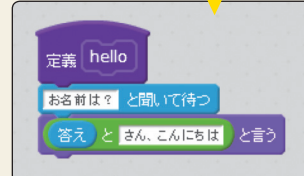
name という名前の変数を用意して、 杉浦 という文字列を代入する

JavaScript では... var name = "杉浦";

Scratch では...



12 Scratchのブロックだと...



※関数や変数に名前をつけるときは、次の3つのことに注意しよう。

- ① 数字、英字、\_と\$の記号が使える
- ② JavaScriptの予約語 (特別な意味を持つ単語) を名前には使えない  
break · case · catch · continue · default · delete · do · else · false · finally · for · function · if · in · instanceof · new · null · return · switch · this · throw · true · try · typeof · var · void · while · with
- ③ 数字から始まる名前をつけることはできない



# クイズアプリを つくろう



ここからは今まで解説したことも使って、プログラミングに関するクイズのアプリをつくってみよう。ぜんぶで4つの画面のアプリだ。



## 1 クイズアプリの設計

まずはこれからつくるアプリの内容を整理しておこう①。アプリを起動すると、クイズが3問出題される。最初の問題はボタンで答える5択の問題だ。次は絵をタップして答える3択の問題。最後は、答えを文字で入力して答えるタイプの問題だよ。

3問の問題に答え終わると、最後のページで正解数が表示され、正解数によってコメントが変わるようにしてみよう。

ここではプログラミングに関するクイズを例にしたけど、問題のテーマはなんでもいいよ！やり方がわかったらオリジナルのクイズアプリをつくってみよう。



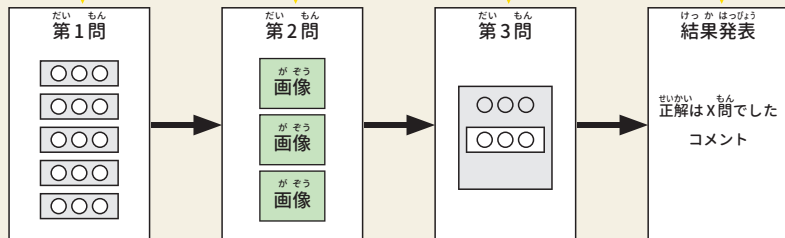
1

ボタンをクリックして答える

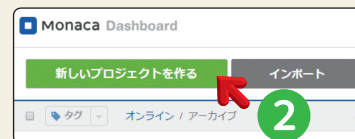
絵をクリックして答える

入力して答える

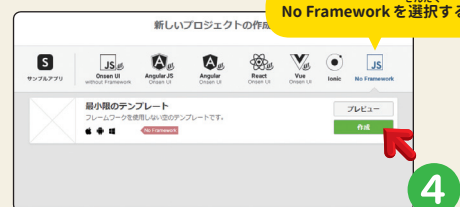
正解数でコメントが変化する



## 2 準備しよう



「新しいプロジェクトを作る」をクリックする②。



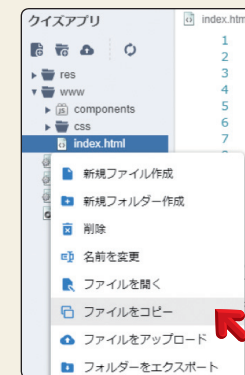
あいさつアプリと同じように、No Frameworkの最小限のテンプレートを③、作成をクリックしよう④。



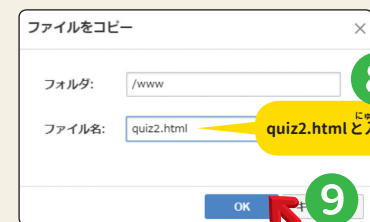
名前はクイズアプリにして、説明もそしてプロジェクトを作成しよう⑤。



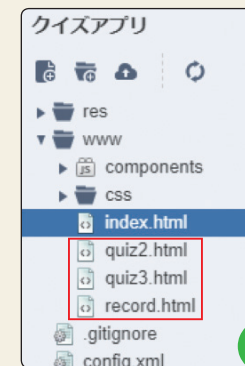
プロジェクトがつくれたら、クラウド IDE で開くをクリックして IDE を起動するよ⑥。



最初にクイズアプリに必要な HTML のファイルを⑦、画面左のプロジェクトパネルの index.html を右クリックし、「ファイルをコピー」を選択しよう⑦。



ファイル名には quiz2.html (2 問目を出題するページ) と入力して⑧、OK をクリックしよう⑨。



おなじように index.html をコピーする作業をして quiz3.html (3 問目を出題するページ) と record.html (結果を表示するページ) をつくろう⑩。



### 3 HTMLの編集と正解判定

まずは1ページ目 (index.html) からつくっていくよ。HTMLを編集して、問題と答えを選ぶボタンを表示させよう<sup>11</sup>。

これはこれまでの復習だから簡単だね。

次にそれぞれのボタンを押したときに実行される関数をつくるよ。正解のボタ

ンは5つのうち1つだけだ。正解だったときに実行される関数をcorrect (正解)、間違っていたときに実行される関数をincorrect (不正解) という名前で行こう<sup>12</sup>。

関数の内容として、正解か不正解かを知らせるためのアラートを表示するようにしておこう。ちなみに、正解 (日本で作られたプログラミング言語) はRubyだ。

次にHTMLのボタンの部分に<sup>12</sup>でくった関数の呼び出しを加えるよ<sup>13</sup>。

Rubyのボタンだけはcorrect関数を呼び出すようにして、それ以外のボタンはみんなincorrect関数を呼び出すようにすればいいね。この段階でボタンを押してみても<sup>14</sup>、正解 (すごい! 正解だよ!) か不正解 (残念! はずれ!) のアラートが表示されることを確認しておこう<sup>15</sup>。

#### 第1問

日本で作られたプログラミング言語は?

Scratch Python Ruby Swift Java

この画面をつくるよ!

```
13 <body>
14 <h1>第1問</h1>
15 <p>日本で作られたプログラミング言語は? </p>
16 <button>Scratch</button>
17 <button>Python</button>
18 <button>Ruby</button>
19 <button>Swift</button>
20 <button>Java</button>
21 </body>
22 </html>
```

```
9 <link rel="stylesheet" href="css/style.css">
10 <script>
11 function correct(){
12   alert("すごい! 正解だよ!");
13 }
14
15 function incorrect(){
16   alert("残念! はずれ!");
17 }
18 </script>
19 </head>
20 <body>
21 <h1>第1問</h1>
22 <p>日本で作られたプログラミング言語は? </p>
```

Rubyのボタンだけcorrect関数を呼び出すようにする

```
19 </head>
20 <body>
21 <h1>第1問</h1>
22 <p>日本で作られたプログラミング言語は? </p>
23 <button onclick="incorrect();">Scratch</button>
24 <button onclick="incorrect();">Python</button>
25 <button onclick="correct();">Ruby</button>
26 <button onclick="incorrect();">Swift</button>
27 <button onclick="incorrect();">Java</button>
28 </body>
29 </html>
```

#### 第1問

日本で作られたプログラミング言語は?

Scratch Python Ruby Swift Java

...144b235d365ab74e5c70be--507299520-8080.monaca-app.com に埋め込まれているページの内容  
すごい! 正解だよ!

14

15

11

12

13

Monaca デバッガーでも同じようにボタン

をタップして<sup>16</sup>、表示ができることを確認しておこう<sup>17</sup>。



### 4 ページを移動する処理

次に、ページを移動するための

処理を追加するよ。今つくっているのは最初の問題を出題するためのページだね。次のページはquiz2.htmlというファイル名でつくってある。正解のときも不正解のときも、次のquiz2.htmlに移動す

るように処理を追加し

よう。location.hrefに移動したいページの名

前を代入すると、ページを移動することができる<sup>18</sup>。

これも実際にボタンを押して、動作確認をしておく

とよいね<sup>19</sup>。まだquiz2.htmlのページは編集をしていないから、ボタンを押して、「This is a template for Monaca app.」と書かれたquiz2.htmlのページが表示されればOKだ。

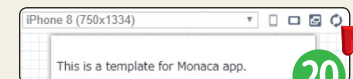
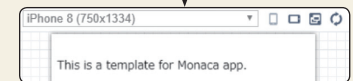
プレビューの表示をindex.htmlに戻したいときは、更新ボタンを押せばいいぞ<sup>20</sup>。Monaca デバッガーの場合は、デバッガーメニューから更新ボタンをタップしよう<sup>21</sup>。



```
9 <link rel="stylesheet" href="css/style.css">
10 <script>
11 function correct(){
12   alert("すごい! 正解だよ!");
13   location.href = "quiz2.html";
14 }
15
16 function incorrect(){
17   alert("残念! はずれ!");
18   location.href = "quiz2.html";
19 }
20 </script>
21 </head>
22 <body>
```



第1問  
日本で作られたプログラミング言語は?  
Scratch Python Ruby Swift Java  
残念! はずれ!



20



21



## 正解数のスコア記録

1 ページ目の製作の仕上げとして、正解数のスコアを記録する方法について解説するね。score という変数をつくって、そこに正解数を保存すればよいのではないかなと思うよね？ 基本的な考え方はその通りだけど、このアプリではぜんぶで4 ページのHTMLを用意して、順番に表示していく。たとえばindex.htmlのページでscoreという変数を用意して、scoreに数字を代入しても、次のquiz2.htmlのページではこのscoreに代入されている値を参照する（読み出す）ことはできない<sup>22</sup>。

そこで、「セッションストレージ」というしくみを使って正解数のスコアを保存することにするよ。このしくみを使うと、開いたページをまたいで変数を使うことができるぞ。使い方は変数とほとんど同じだけど、少しだけ書き方が違うよ。ここではセッションストレージの中にscoreという変数をつくって、そこに正解数を保存するよ<sup>23</sup>。

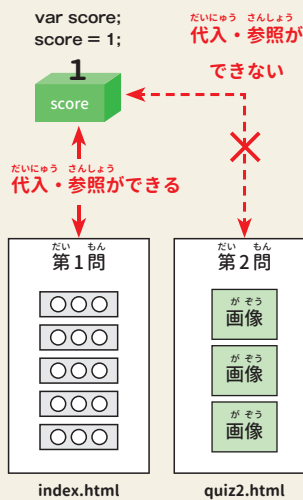
正解の場合は1を、不正解の場合は0を保存しておく<sup>24</sup>。

```

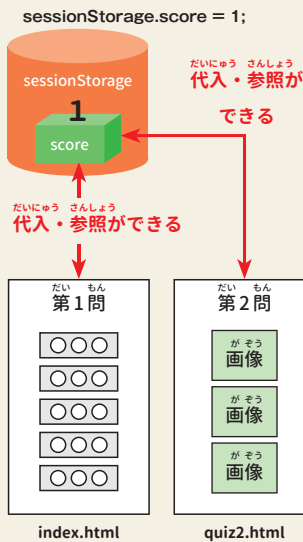
9 <link rel="stylesheet" href="css/style.css">
10 <script>
11 function correct(){
12   alert("すごい！正解だよ！");
13   location.href = "quiz2.html";
14   sessionStorage.score = 1;
15 }
16
17 function incorrect(){
18   alert("残念！はずれ！");
19   location.href = "quiz2.html";
20   sessionStorage.score = 0;
21 }
22 </script>

```

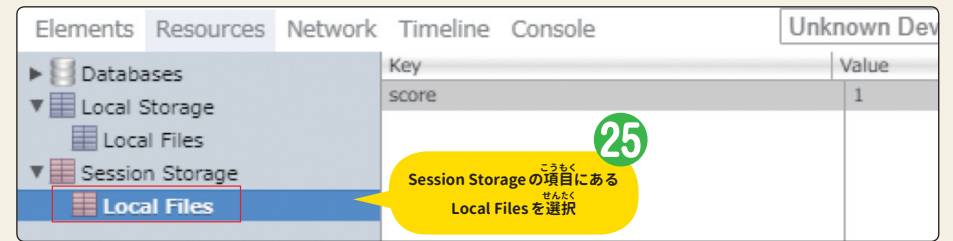
22



23



Monaca デバッガーを使っているときは、デバッガーのResourcesというタブからセッションストレージの内容を確認することができるぞ<sup>25</sup>。



最初のページ (index.html) のHTML と JavaScript を<sup>26</sup>にまとめておくね。

```

1 <!DOCTYPE HTML>
2 <html>
3 <head>
4   <meta charset="utf-8">
5   <meta name="viewport" content="width=device-width, initial-scale=1, maximum-scale=1,
6   <meta http-equiv="Content-Security-Policy" content="default-src * data: gap: content:
7   <script src="components/loader.js"></script>
8   <link rel="stylesheet" href="components/loader.css">
9   <link rel="stylesheet" href="css/style.css">
10  <script>
11    function correct(){
12      alert("すごい！正解だよ！");
13      location.href = "quiz2.html";
14      sessionStorage.score = 1;
15    }
16
17    function incorrect(){
18      alert("残念！はずれ！");
19      location.href = "quiz2.html";
20      sessionStorage.score = 0;
21    }
22  </script>
23 </head>
24 <body>
25   <h1> 第1問 </h1>
26   <p> 日本で作られたプログラミング言語は？ </p>
27   <button onclick="incorrect();">Scratch</button>
28   <button onclick="incorrect();">Python</button>
29   <button onclick="correct();">Ruby</button>
30   <button onclick="incorrect();">Swift</button>
31   <button onclick="incorrect();">Java</button>
32 </body>
33 </html>

```





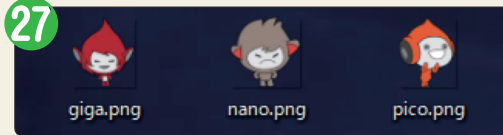
## 6 画像の準備

次は2ページ目 (quiz2.html) をつ  
くってこよう。まずはクイズの画像を  
用意しておこう。今回はScratchに登  
場する3つのオリジナルキャラクター  
の名前 (ギガ、ナノ、ピコ) を当てるク  
イズにしたよ。用意する画像のファ  
イル名は半角英数字でつけておこう<sup>27</sup>。  
これらの画像の保存場所はデスクト  
ップにしておこう。

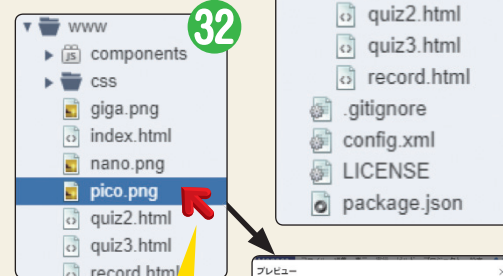
この画像ファイルをMonacaで使う  
には、ファイルをアップロードする必  
要がある。IDEの画面の左にあるプロ  
ジェクトパネルから、wwwという  
フォルダーのアイコンを右クリックし  
てメニューを表示させ、「ファイルを  
アップロード」を選択しよう<sup>28</sup>。表示  
されたウィンドウの真ん中の部分に、  
デスクトップにある画像ファイルをド  
ラッグするか、ファイル選択ボタンを  
押して画像ファイルを選択して、3枚  
ともアップロードしよう<sup>29</sup>。

アップロードが終わったら、×ボタンでウィンドウは閉じておこう<sup>30</sup>。

プロジェクトパネルに3つのファイルが追加されているか確認しよう<sup>31</sup>。ファイルを  
ダブルクリックすれば、画像を見ることもできるよ<sup>32</sup>。



3つの画像ファイルを追加



ダブルクリックすると、  
画像ファイルが確認できる



## 7 問題文と画像表示のタグ

```
12 </head>
13 <body>
14 <h1>第2問</h1>
15 <p>Scratchのキャラクター、ピコは? </p>
16 
17 
18 
19 </body>
20 </html>
```

ファイルがアップロードできたら、問題文と画像を表示  
するためのタグをquiz2.htmlに書き加えよう。まずはquiz2.  
htmlをダブルクリックして、エディターを開き、3つのimg  
タグを加えるよ<sup>33</sup>。この画像の表示の方法はPart 4で解説し  
た内容だから、大丈夫だね。

プレビューでは、常にindex.htmlの内容が表示されるよ  
うになっている。今つくっているquiz2.htmlの画面を確認  
したい場合は、第1問に回答をして、次の第2問に進めば  
quiz2.htmlの内容を確認することができるぞ。

スマホは画面が小さいので、一列に3つの画像が表示され  
るように、画像の横幅を90px (ピクセル) に指定したよ。  
widthの前に半角でスペースを忘れないように注意しよう。  
使っているスマホの画面が狭い場合は、90pxという数値を  
変える必要がある。実際にMonacaデバッガーで確認してみ  
るといいぞ<sup>34</sup>。

### 第2問

Scratchのキャラクター、ピコは?



### 第2問

Scratchのキャラクター、ピコは?





## 8 関数をコピーする

次に関数を用意しよう。じつは関数の中身は index.html でつくった JavaScript と同じだから、index.html でつくった 2 つの関数 (correct 関数と incorrect 関数) の文字をドラックで選択して、Ctrl キーと C のキー (Mac の場合は Command キーと C のキー) を同時に押してコピーしよう 35。そのまま quiz2.html に戻って、11 行目にカーソルを合わせて、Ctrl キーと V のキー (Mac の場合は Command キーと V のキー) を同時に押して貼りつけよう 36。タブを使って 2 つのファイルを切り替えることができるぞ。貼りつけたときにインデントがずれたら修正しておこう。

index.html のタブを選択

Ctrl キーと C のキーを同時に押してコピーする

quiz2.html のタブを選択

11 行目にカーソルを合わせて、Ctrl キーと V のキーを同時に押して貼り付ける



## 9 2 ページ目の仕上げ

correct 関数と incorrect 関数の内容を少しだけ変更しよう 37。修正のポイントは 2 つだよ。1 つ目は、移動するページの名前の変更だ。両方とも quiz3.html に移動するように変更しておこう。

2 つ目は、スコアの計算。正解の場合は、第 1 問で獲得したスコアに、第 2 問で獲得したスコアを足す必要がある。不正解の場合は、スコアは 0 から変化しないから、incorrect 関数のスコアに関する命令は消しておこう。

現在の score に 1 を加える

Scratch のブロックだと...

次に画像をタップしたら実行する関数を指定しよう。ボタンと同じように onclick を追加すれば、画像をクリックしたときに実行する関数を指定することができるぞ。ピコ

の画像 (pico.png) だけは correct 関数を実行するようにして、あとの 2 つは incorrect

関数を指定すれば

よい 38。

```
22 </head>
23 <body>
24 <h1>第2問</h1>
25 <p>Scratchのキャラクター、ピコは？</p>
26 
27 
28 
29 </body>
30 </html>
```

39 に第 2 問のページ (quiz2.html) の HTML と JavaScript をまとめておくね。プレビューでも動作を確認して、第 3 問のページに移動することをチェックしておこう。

```
1 <!DOCTYPE HTML>
2 <html>
3 <head>
4 <meta charset="utf-8">
5 <meta name="viewport" content="width=device-width, initial-scale=1, maximum-scale=1,
6 <meta http-equiv="Content-Security-Policy" content="default-src * data: gap: content:
7 <script src="components/loader.js"></script>
8 <link rel="stylesheet" href="components/loader.css">
9 <link rel="stylesheet" href="css/style.css">
10 <script>
11     function correct(){
12         alert("すごい！正解だよ！");
13         location.href = "quiz3.html";
14         sessionStorage.score++;
15     }
16
17     function incorrect(){
18         alert("残念！はずれ！");
19         location.href = "quiz3.html";
20     }
21 </script>
22 </head>
23 <body>
24 <h1>第2問</h1>
25 <p>Scratchのキャラクター、ピコは？</p>
26 
27 
28 
29 </body>
30 </html>
```

## 10 3 ページ目の HTML の編集

```

12 </head>
13 <body>
14   <h1>第3問</h1>
15   <p>JavaScriptで変数を作るときに使う命令は? </p>
16   <button onclick="quiz3();">答えを入力</button>
17 </body>
18 </html>

```

40

第3問のページをつくっていきましょう。まずはHTMLからだね。問題文と、答えを入力するためのボタンを表示するようにしたよ<sup>40</sup>。ボタンを押して実行する関数はquiz3という名前にしたぞ。

### 第3問

JavaScriptで変数を作るときに使う命令は？

答えを入力

このボタンを追加

## 11 入力を受け取り

```

9   <link rel="stylesheet" href="css/style.css">
10  <script>
11    function quiz3(){
12      var answer = prompt("ヒント：英語で3文字だよ");
13      if(answer == "var"){
14        alert("すごい！正解だよ！");
15      }else{
16        alert("残念！はずれ！");
17      }
18    }
19  </script>
20 </head>
21 <body>
22   <h1>第3問</h1>
23   <p>JavaScriptで変数を作るときに使う命令は? </p>
24   <button onclick="quiz3();">答えを入力</button>

```

41

次はquiz3の関数をつくっていきましょう。プロンプトを表示して答えを入力してもらい、それが正解かどうかを調べるJavaScriptを書いていくぞ<sup>41</sup>。プロンプトを表示して変数に代入するのは、あいさつアプリの応用だ。今回の変数名はanswer（答え）にしたよ。これで入力された内容がanswerという変数に代入できる。そのanswerの内容を調べて、正解か不正解かを判断する必要があるね。

### 条件分岐の書き方

```

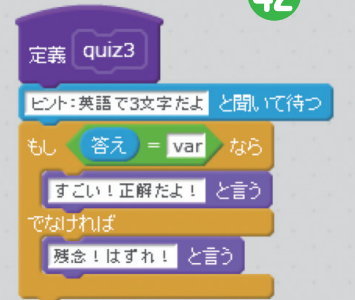
if(条件){
  // 条件が成立するの処理
}
else{
  // 条件が不成立のときの処理
}

```

{ }を対応させ、この間に条件の成立・不成立のそれぞれの場合にしたい処理を書く

Scratchのブロックだと...

42



JavaScriptではifという命令を使って「条件分岐」(41ページ参照)を書くぞ。上に条件分岐の書き方をまとめたので参考にしよう。Scratchのブロックと比べてみたのが<sup>42</sup>だ。Scratchと違うのは、入力された答えを保存しておく変数answerを自分で用意していることと、答えを調べるときの記号が==になっていることだね。正解のvarという文字列を入力すると、正解と表示されること。違う文字列を入力すると不正解となることを確認しておこう<sup>43</sup>。

43

正解のvarを入力

不正解のhenを入力

...007820a206b46a84ad039d--507299523-8080.monaca-app.com に埋め込まれているページの内容

ヒント：英語で3文字だよ

var

OK キャンセル

...007820a206b46a84ad039d--507299523-8080.monaca-app.com に埋め込まれているページの内容

ヒント：英語で3文字だよ

hen

OK キャンセル

...007820a206b46a84ad039d--507299523-8080.monaca-app.com に埋め込まれているページの内容

すごい！正解だよ！

OK

...007820a206b46a84ad039d--507299523-8080.monaca-app.com に埋め込まれているページの内容

残念！はずれ！

OK





## 12 3 ページ目の仕上げ

正解の場合も不正解の場合も、最後のページの record.html に移動する。また、正解の場合はスコアを増やすという処理を追加しよう 44。

45 に第 3 問のページ (quiz3.html) の HTML と JavaScript をまとめておくね。

```

9 <link rel="stylesheet" href="css/style.css">
10 <script>
11 function quiz3(){
12   var answer = prompt("ヒント：英語で3文字だよ");
13   if(answer == "var"){
14     alert("すごい！正解だよ！");
15     location.href = "record.html";
16     sessionStorage.score++;
17   }else{
18     alert("残念！はずれ！");
19     location.href = "record.html";
20   }
21 }
22 </script>
23 </head>
24 <body>
25 <h1>第3問</h1>
26 <p>JavaScriptで変数を作るときに使う命令は？</p>

```

44

45

```

1 <!DOCTYPE HTML>
2 <html>
3 <head>
4   <meta charset="utf-8">
5   <meta name="viewport" content="width=device-width, initial-scale=1, maximum-scale=1,
6   <meta http-equiv="Content-Security-Policy" content="default-src * data: gap: content:
7   <script src="components/loader.js"></script>
8   <link rel="stylesheet" href="components/loader.css">
9   <link rel="stylesheet" href="css/style.css">
10  <script>
11    function quiz3(){
12      var answer = prompt("ヒント：英語で3文字だよ");
13      if(answer == "var"){
14        alert("すごい！正解だよ！");
15        location.href = "record.html";
16        sessionStorage.score++;
17      }else{
18        alert("残念！はずれ！");
19        location.href = "record.html";
20      }
21    }
22  </script>
23 </head>
24 <body>
25   <h1>第3問</h1>
26   <p>JavaScriptで変数を作るときに使う命令は？</p>
27   <button onclick="quiz3();">答えを入力</button>
28 </body>
29 </html>

```

追加した  
JavaScript

追加した  
HTML



## 13 再挑戦ボタンの作成

いよいよ最後の結果表示 (record.html) のページだ。

まずはもう一度クイズに挑戦できるようにするためのボタンを用意して、そのボタンが押されたら実行される reset という名前の関数をつくろう。reset の関数では、第 1 問を出題する index.html に戻るようにして、スコアを 0 に戻しておくよ 46。

```

9 <link rel="stylesheet" href="css/style.css">
10 <script>
11 function reset(){
12   location.href = "index.html";
13   sessionStorage.score = 0;
14 }
15 </script>
16 </head>
17 <body>
18   <button onclick="reset();">もう一度最初から</button>
19 </body>
20 </html>

```

46



## 14 コメント表示

次はスコアに応じたコメントを表示する部分をつくろう。関数として

```

9 <link rel="stylesheet" href="css/style.css">
10 <script>
11 document.write("<h1>結果発表</h1>");
12
13 function reset(){
14   location.href = "index.html";
15   sessionStorage.score = 0;
16 }
17 </script>

```

47

つくった処理は、ページが読み込まれたときには実行されず、関数を呼び出すことが必要だったね。script のタグに書かれた JavaScript の命令は関数としてつくった以外の部分の命令は、ページが読み込まれたときに実行される。最初に結果発表という文字を画面に表示する命令をつけ加えて、動作を確認してみよう 47。追加するのは、document.write という命令だよ。これは画面に指定した文字列を表示するものだ。この命令は reset 関数の上に書いてあるぞ。script タグの中に書かれた命令は基本的には上から順番に実行されていくけれど、関数としてつくった部分は呼び出しがなければ実行はされない

48。

48

```

9 <link rel="stylesheet" href="css/style.css">
10 <script>
11 document.write("<h1>結果発表</h1>");
12
13 function reset(){
14   location.href = "index.html";
15   sessionStorage.score = 0;
16 }
17 </script>
18 </head>
19 <body>
20   <button onclick="reset();">もう一度最初から</button>

```

ページが読み込まれたら実行

reset 関数の呼び出しがあるまで実行されない

```

9 <link rel="stylesheet" href="css/style.css">
10 <script>
11 document.write("<h1>結果発表</h1>");
12
13 function reset(){
14   location.href = "index.html";
15   sessionStorage.score = 0;
16 }
17 </script>
18 </head>
19 <body>
20 <button onclick="reset();">もう一度最初から</button>

```

49

JavaScriptで表示した文字

結果発表

もう一度最初から

HTMLで表示したボタン

scriptタグの後のHTMLも上から順番に表示されていくから、今回は結果発表という文字はJavaScriptで表示させて、ボタンはHTMLで表示させているということになるね49。

なぜこんなことをしているかというと、結果発表という表示は毎回共通だけど、スコアはsessionStorage.scoreに保存してあり、コメントはその値によって変化をさせる必要があるからだ。HTMLにスコアに関するコメントを書いてしまっただけでは、毎回同じ表示になってしまう。だからJavaScriptからコメントを表示させる必要があるんだね。

```

9 <link rel="stylesheet" href="css/style.css">
10 <script>
11 document.write("<h1>結果発表</h1>");
12 document.write("<h2>正解は" + sessionStorage.score + "問でした</h2>");
13 if(sessionStorage.score == 3){
14   document.write("<p>全問正解！大変よくできました！</p>");
15 }else{
16   document.write("<p>もう一度挑戦して、全問正解を目指そう！</p>");
17 }
18
19 function reset(){
20   location.href = "index.html";
21   sessionStorage.score = 0;
22 }
23 </script>
24 </head>
25 <body>
26 <button onclick="reset();">もう一度最初から</button>

```

50

ではスコアとそれに応じたコメントを表示する部分をつけ加えよう。今回はシンプルに3問正解だった場合とそれ以外の場合に分けてみたよ50。record.htmlについては、HTMLとJavaScriptのまとめはないけれど、50をよく見て確認しよう。クイズアプリでつくるHTMLとJavaScriptは、8ページで紹介しているサポートページからダウンロードもできるよ。うまく動作しないときはサポートページを見てね。



## 15 動作結果をチェック

これでクイズアプリは完成だ。Monacaデバッガーを使って、全問正解の場合51、そうでない場合の両方の動作確認をしておこう。

51



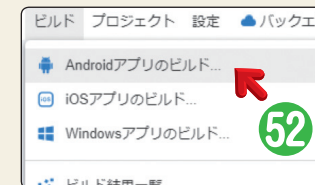
## 16 アプリのビルドの準備

Monacaデバッガーでもアプリの動作を確認することができるけれど、Androidのスマホを使っている場合、ほかのアプリと同じように、クイズアプリとして動作させることができるぞ。必要なファイルを準備して、アプリの準備をすることを「ビルド」というよ。

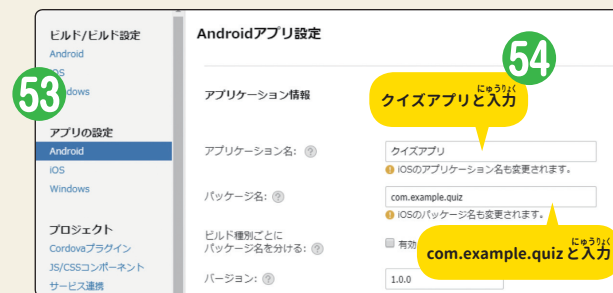
まずはビルドのメニューから「Androidアプリのビルド」を選択しよう52。

アプリの設定のAndroidの項目を選択して53、アプリケーション名とパッケージ名を

変更しよう。パッケージ名は元から入力されている「com.example.helloworld」を少し変更するだけで大丈夫だよ54。



52



54

53



## 17 アイコンの設定

次にアプリのアイコンを設定しよう。アイコンにしたい画像をpng形式で用意しよう。ある程度の大きさの画像を用意して、一括で設定するのがある「アップロード」を選択し、アイコンとして設定したい画像ファイルを選択しよう。

クイズアプリなので、はてなマークの画像をアップしているよ。アイコンの設定が終わったら「保存する」をクリックしよう。



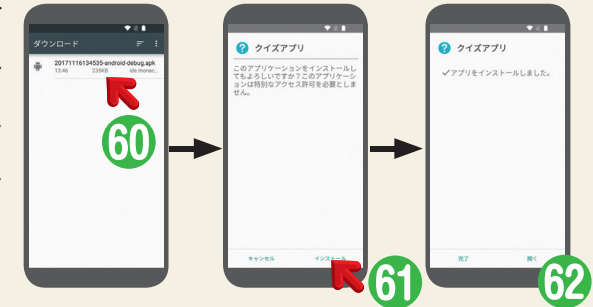
## 18 ビルドの開始とインストール

画面の左から、ビルド/ビルド設定のAndroidを選択しよう。デバッグビルドを選択して、「ビルドを開始する」のボタンを押すよ。新しい画面が現れて処理が始まるので、しばらく待とう。

ビルドがうまくいくと、QRコードなどが表示されたページに切り替わるぞ。一番手軽なのは、画面に表示されたQRコードをスマホで読み取ってアプリをインストールする方法だ。



QRコードを読み取れば、アプリのファイルがダウンロードできるよ。ダウンロードしたファイルを選択して、インストールをタップしよう。ダウンロードの際に、Monacaのサイトにサインインするように指示された場合は、メールアドレスとパスワードを入力しよう。「アプリをインストールしました。」の表示ができれば、アプリのインストールは完了だね。



ほかのアプリと同じようにクイズアプリが使えるようになるぞ。なお、今使っている無料のプランだと、1日に3回までしかビルドができないから注意しよう。

iPhoneなどのiOSが動作しているスマートフォンについては、ビルドにアップル社から発行してもらう証明書が必要なので、少し準備が大変だよ。詳しい方法については以下のMonacaのマニュアルを見てみよう。

[https://docs.monaca.io/ja/monaca\\_ide/manual/build/ios/build\\_ios/](https://docs.monaca.io/ja/monaca_ide/manual/build/ios/build_ios/)



スマホアプリをつくるのはどうだったかな？  
シンプルなクイズアプリをつくってみたけれど、いろいろ改造して、オリジナルのクイズアプリにしてみよう！

※アプリがインストールできない場合は、「設定」のメニューから「ユーザー設定」の「セキュリティ」を選択し、「提供元不明のアプリ」を一時的に許可しよう。インストールが完了したら「提供元不明のアプリ」の設定は元に戻すようにすること。