6章 条件分岐のif文を理解しよう

ある条件によって実行する処理を切り替える機能を学びましょう。

120分 - 読了

6.1 本章の目標

本章では以下を目標にして学習します。

- 条件分岐とは何か、概要をつかむこと
- if文の書き方を知り、実際にコードを書いてみること
- 「〜より大きいならば（ > ）」「〜より小さいならば（ < ）」など、条件分岐に使う比較演算子の種類を知ること

プログラミングをしていると、「ある条件に当てはまるときだけこの処理を実行したい」という場面がたくさん出てきます。

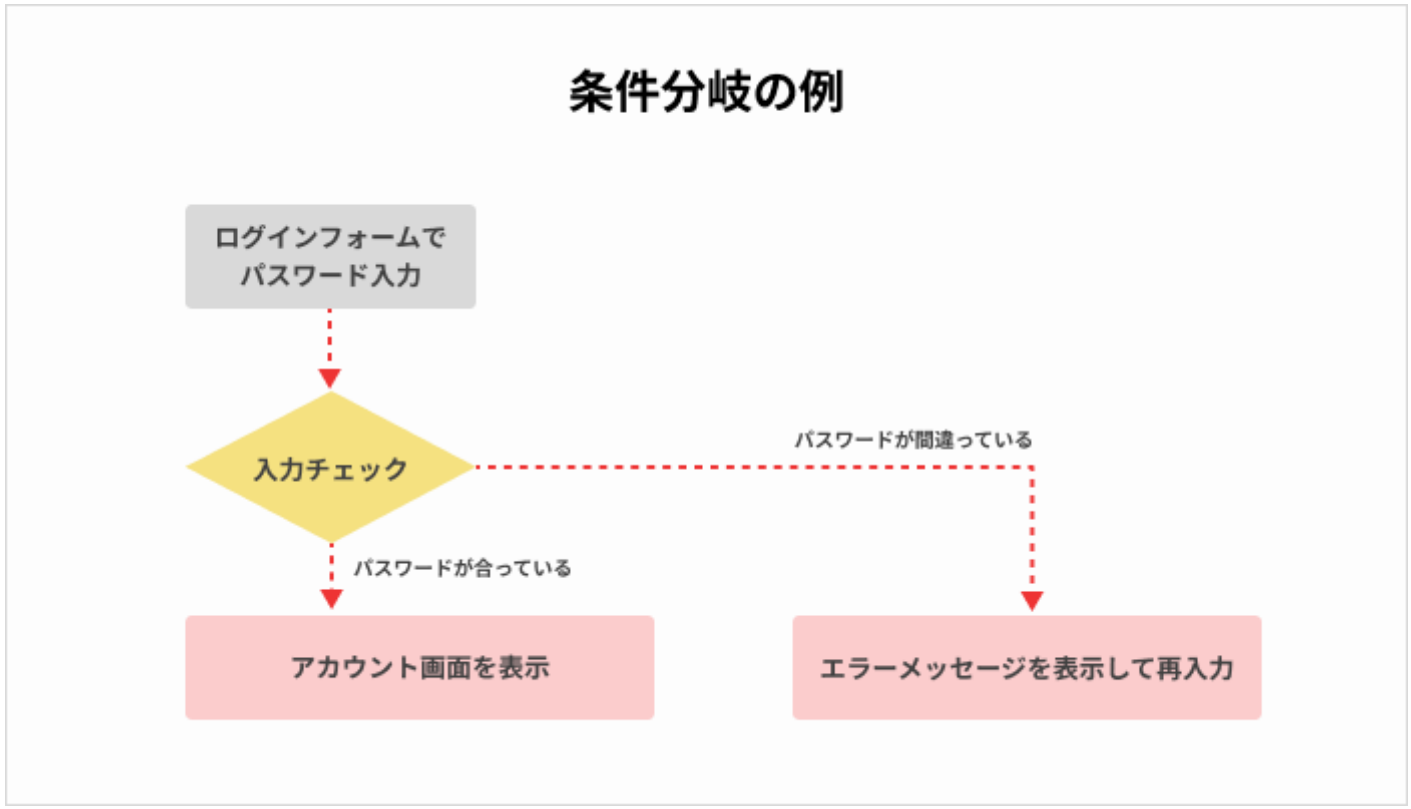
そこで使うのが、**条件分岐**と呼ばれる処理です。

条件分岐を使うことで、「もし金曜日なら商品を割引する」「未ログインであればログイン画面を表示する」など、さまざまな機能を実装できるようになります。

これまでの章よりも難易度は上がりますが、条件分岐はJavaScriptをはじめとするPHP・Ruby・Pythonなどのプログラミング言語でよく使う重要な処理です。最初にマスターすべき処理といっても過言ではありません。ここでしっかりと覚えておきましょう。

6.2 条件分岐とは

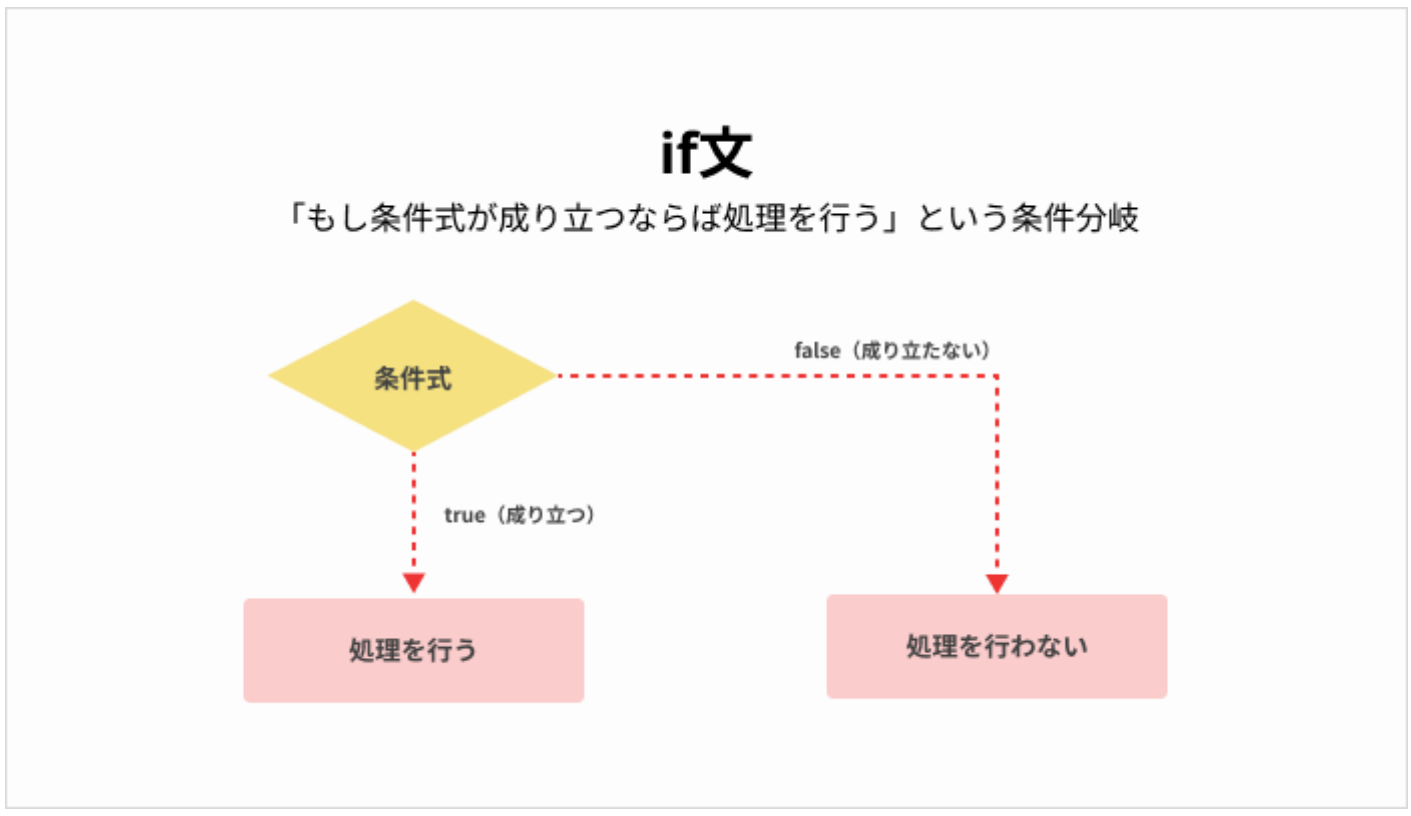
条件分岐とは、「必須事項が入力されていたらフォーム内容を送信し、未入力であればエラーメッセージを表示する」など、**条件によって処理を分けること**です。



JavaScriptで条件分岐を行うときは、**if文**または**switch文**を使います。本章ではif文について学び、次章でswitch文について学びます。

現時点では、分岐が少ないときはif文、分岐が多いときはswitch文を使うという認識でOKです。

6.3 if文の書き方



if文は「もし条件式が成り立つならば、処理を行う」という条件分岐です。簡単にいえば、「○○の場合、□□する」ということです。

if文は以下のように書きます。5節で実際にコードを書くので、現時点ではイメージだけつかめればOKです。

```
if（条件式）{
  条件が成り立つときの処理
}

```

丸括弧 () の中に条件式を記述し、波括弧 {} の中に条件が成り立つときの処理を記述します。

なお、条件が成り立つかどうかを判定する式のことを、JavaScriptをはじめとするプログラミング言語では**条件式**といいます。条件式は条件が成り立てば `true` を返し、成り立たなければ `false` を返します。

そして、 `true` が返されたときに処理が実行されます。

if文の使用例

if文の使用例を見てみましょう。以下の `num > 10` の部分が条件式です。以下の例ではif文の前に `const num = 50;` と記述しているため、`num > 10` という条件が成り立ち（ `true` が返され）、「 `定数numは10より大きいです` 」という文字列が出力されます。

- num＝「番号」や「数字」を意味するnumberの略で、慣習的に使われる変数名

なお、条件式に使う `>` などの記号を**比較演算子**といいます。

JSファイル（見本）

```
const num = 50;

// もし定数numが10より大きいならば、「定数numは10より大きいです」という文字列を出力する
if (num > 10) {
  console.log('定数numは10より大きいです');
}

```

「trueを返す」とはどういうことか（戻り値について）

比較演算子を使って2つの値を比較することで、 `true` （真）または `false` （偽）のいずれかの値（3章で学んだ論理型のデータ）が返されます。そして、その返ってきた値が `true` であれば処理を行うのがif文です。

この「返ってきた値」のことを、プログラミングでは**返り値**または**戻り値**といいます（本教材では「戻り値」で説明します）。四則演算の計算結果なども、戻り値の1つです。

戻り値をコンソールに出力してみよう

実際にコードを書いて確かめてみましょう。

まずはVisual Studio Codeを開き、 `js` フォルダ内に新しく `if-switch.js` というファイルを作成してください。

続いて、 `if-switch.js` を以下のように編集してください。違いを比べるために、3章で学んだ算術演算子も使っています。

if-switch.js

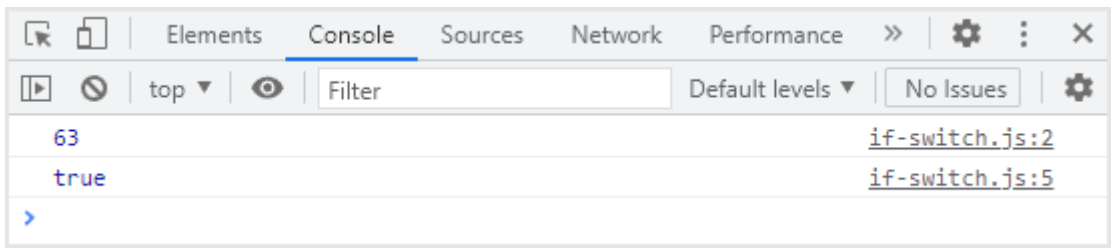
```
// 算術演算子を使った場合の戻り値を出力する
console.log(45 + 18);

// 比較演算子を使った場合の戻り値を出力する
console.log(45 > 18);

```

次に `index.html` を以下のように編集し、読み込むJSファイルを `if-switch.js` に変更してください。

index.html

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

<head>
  <meta charset="UTF-8">
  <title>JavaScript基礎編</title>
</head>

<body>
-   <script src="js/constant.js"></script>
+   <script src="js/if-switch.js"></script>
</body>

</html>
```

`index.html` をブラウザで開き、デベロッパーツールのコンソールを確認してみましょう。以下のように、それぞれの戻り値が表示されていればOKです。



算術演算子の場合は計算結果（数値型のデータ）が戻り値として返され、比較演算子の場合は `true` または `false` （論理型のデータ）が戻り値として返されることがおわかりいただけたかと思います。

6.4 条件式に使う比較演算子

前述のとおり、 条件式には `>` などの比較演算子を使います。

条件式に使う主な比較演算子は以下のとおりです。実際に使いながら覚えていけばよいので、ここではざっと確認しましょう（等価演算子 `==` と厳密等価演算子 `===` の違いについては後述します）。

比較演算子	処理の内容
`==`	2つの値が等しい場合は `true` を返す（等価演算子）。
`===`	2つの値とデータ型が等しい場合は `true` を返す（厳密等価演算子）。
`!=`	2つの値が等しくない場合は `true` を返す。
`!==`	2つの値とデータ型が等しくない場合は `true` を返す。
`>`	左辺の値が右辺の値よりも大きい場合は `true` を返す。
`>=`	左辺の値が右辺の値以上の場合は `true` を返す。
`<`	左辺の値が右辺の値よりも小さい場合は `true` を返す。
`<=`	左辺の値が右辺の値以下の場合は `true` を返す。

等価演算子と厳密等価演算子の違い

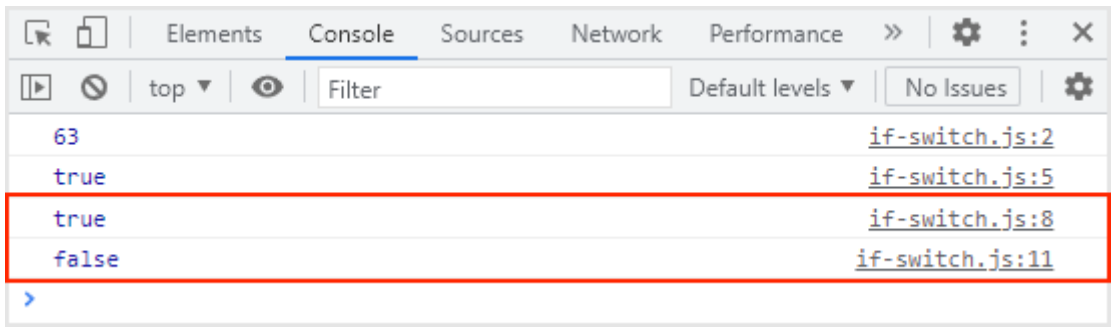
比較演算子の中で特に注意したいのが、 == と === の違いです。 == を**等価演算子**といい、 === を**厳密等価演算子**といいます。いずれも2つの値が等しいかどうかを確かめる比較演算子ですが、処理の内容は少し異なります。

実際にコードを書いて違いを比べてみましょう。 if-switch.js を以下のように編集してください。以下のコードでは、文字列型の '5' と数値型の 5 を比較しています。

if-switch.js

```
// 算術演算子を使った場合の戻り値を出力する
console.log(45 + 18);

// 比較演算子を使った場合の戻り値を出力する
console.log(45 > 18);

+ // 等価演算子を使った場合の戻り値を出力する
+ console.log('5' == 5);
+
+ // 厳密等価演算子を使った場合の戻り値を出力する
+ console.log('5' === 5);

```

では index.html をブラウザで開き、デベロッパーツールのコンソールを確認してみましょう。以下のように、等価演算子 == と厳密等価演算子 === で戻り値が異なります。



値としてはどちらも「5」を意味しているため、 == を使った比較ではどちらの値も等しいという解釈になり、 true が返されます。

一方で === を使った比較では、値が同じでも文字列型と数値型というデータ型の違いがあるため、 false が返されます。

つまり、 === を使えば値とデータ型を一緒に比較してくれるということです。より厳密に比較できるため、基本的には厳密等価演算子である === を使うようにしましょう。

最後にもう一度、等価演算子 == と厳密等価演算子 === の違いを掲載しておきます。

- == （等価演算子）：2つの値が等しい場合は true を返す
- === （厳密等価演算子）：2つの値とデータ型が等しい場合は true を返す

ここまでで前半戦が終了です。次からはいよいよ、実際にif文を書いていきます。ただしその前に、一度休憩しておきましょう。

6.5 if文を書いてみよう

では、実際にif文を書いてみましょう。以下の3パターンに分けて、簡単なプログラムを作成します（ `else` と `else if` については後述します）。

1. `if` のみを記述するパターン（もし○○であれば、●●する）
2. `if` と `else` を記述するパターン（もし○○であれば●●し、それ以外であれば▲▲する）
3. `if` 、 `else if` 、 `else` を記述するパターン（もし○○であれば●●し、そうでなくて□□であれば■■し、それ以外であれば▲▲する）

順番にやっていきましょう。

1. ifのみを記述するパターン（もし○○であれば、●●する）

まずは、「値が `4` であれば『 `大当たりです` 』という文字列を出力する」プログラムを作成します。作成手順は以下のとおりです。

1. 変数 `num` に `0` 〜 `4` までのランダムな整数を代入する
2. 変数 `num` の値が `4` であれば、「 `大当たりです` 」という文字列を出力する

覚える必要はありませんが、今回はランダムな整数を生成するために `Math.floor(Math.random() * n)` というコードを記述します。 `Math.floor(Math.random() * n)` は、 `0` 〜 `n - 1` までのランダムな整数を生成するコードです。

詳しくは応用編で解説するので、現時点では「数字を指定しなくてもランダムな数字を表示してくれるおまじない」という認識でOKです。

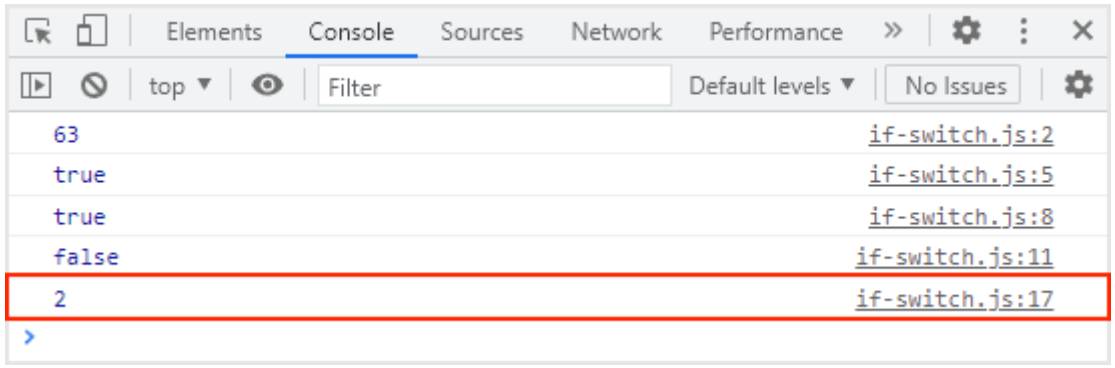
では、 `if-switch.js` を以下のように編集してください。

`if-switch.js`

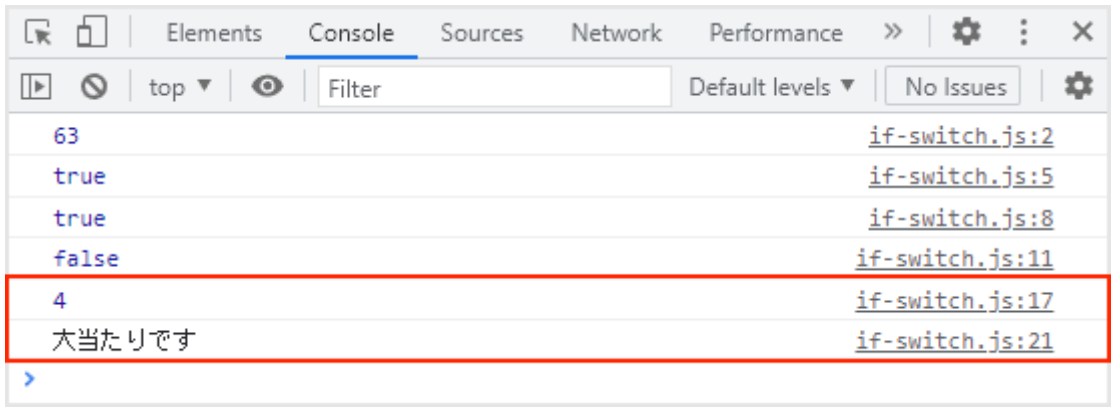
```
// 算術演算子を使った場合の戻り値を出力する
console.log(45 + 18);

// 比較演算子を使った場合の戻り値を出力する
console.log(45 > 18);

// 等価演算子を使った場合の戻り値を出力する
console.log('5' == 5);

// 厳密等価演算子を使った場合の戻り値を出力する
console.log('5' === 5);

+ // 変数numに0〜4までのランダムな整数を代入する
+ let num = Math.floor(Math.random() * 5);
+
+ // 変数numの値を出力する（確認用）
+ console.log(num);
+
+ // 変数numの値が4であれば、「大当たりです」という文字列を出力する
+ if (num === 4) {
+   console.log('大当たりです');
+ }
```

`index.html` をブラウザで開き、デベロッパーツールのコンソールを確認してみましょう。変数 `num` の値が `4` になるまでブラウザを更新してください。

変数 `num` の値が `4` 以外のときは、以下のように何も表示されません。



しかし、変数 `num` の値が `4` のときは、以下のように「 `大当たりです` 」という文字列が表示されます。



2. ifとelseを記述するパターン（もし○○であれば●●し、それ以外であれば▲▲する）

続いて、以下のようなプログラムを作成します。

1. 変数 `num` に `0` 〜 `4` までのランダムな整数を代入する
2. 変数 `num` の値が `4` であれば、「 `大当たりです` 」という文字列を出力する
3. それ以外のときは、「 `はずれです` 」という文字列を出力する

注目していただきたいのは、3つ目の「それ以外のとき」です。これまでは `if` のみを記述し、「それ以外のとき」、つまり条件式が成り立たないときは処理を何も行いませんでした。

しかし、「それ以外のとき」にも何らかの処理を実行したい場合もあります（今回の例では「 `はずれです` 」という文字列を出力する）。

そこで記述するのが `else` です。以下のように `else` を記述することで、条件式が成り立たないときにも処理を行うことができます。

```
if（条件式）{
  条件が成り立つときの処理
}
else {
  条件が成り立たないときの処理
}
```

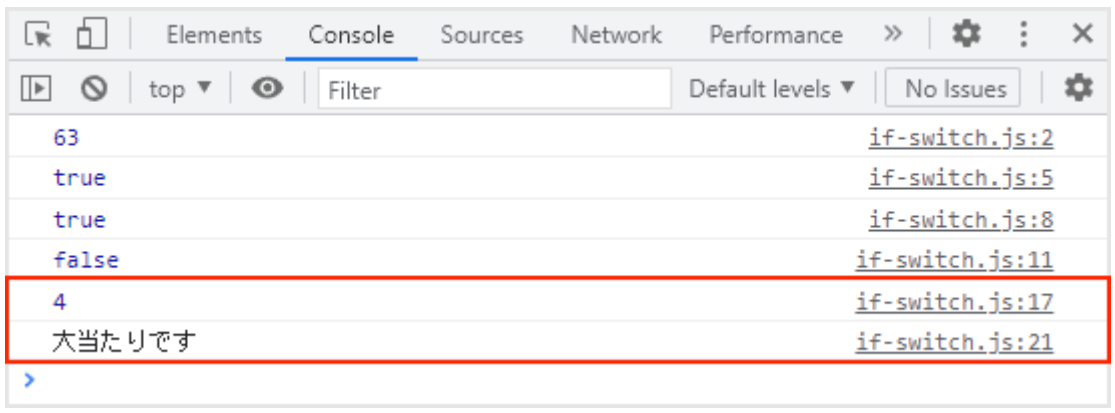
では実際にやってみましょう。 `if-switch.js` を以下のように編集してください。

if-switch.js

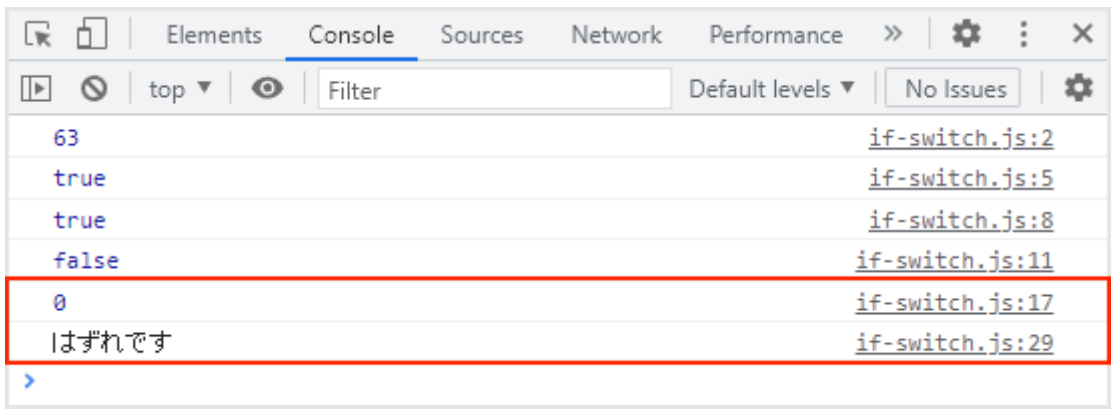
```
//========= 前略 ========

// 変数numに0〜4までのランダムな整数を代入する
let num = Math.floor(Math.random() * 5);

// 変数numの値を出力する（確認用）
console.log(num);

// 変数numの値が4であれば、「大当たりです」という文字列を出力する
if (num === 4) {
  console.log('大当たりです');
}
+ // それ以外のときは、「はずれです」という文字列を出力する
+ else {
+   console.log('はずれです');
+ }

```

`index.html` をブラウザで開き、デベロッパーツールのコンソールを確認してみましょう。変数 `num` の値が `4` のときは「 `大当たりです` 」という文字列が表示され、それ以外のときは「 `はずれです` 」という文字列が表示されていればOKです。

変数numの値が4のとき



変数numの値がそれ以外（0、1、2、3）のとき



3. if、else if、elseを記述するパターン（もし○○であれば●●し、そうでなくて□□であれば■■し、それ以外であれば▲▲する）

最後に、以下のようなプログラムを作成します。

1. 変数 `num` に `0` 〜 `4` までのランダムな整数を代入する
2. 変数 `num` の値が `4` であれば、「 `大当たりです` 」という文字列を出力する
3. 変数 `num` の値が `3` であれば、「 `当たりです` 」という文字列を出力する
4. それ以外のときは、「 `はずれです` 」という文字列を出力する

注目していただきたいのは、3つ目の処理です。このように、条件式を複数作りたい場合は `else if` を記述します。

```
if（条件式A）{
  条件Aが成り立つときの処理
}
else if（条件式B）{
  条件Bが成り立つときの処理
}
else {
  どの条件も成り立たないときの処理
}

```

なお、 `if` と `else` は1つの条件分岐の中で一度しか記述できませんが、 `else if` はいくつでも追加できます。ただし、 `else if` が多すぎるとコードが見づらくなってしまうので、コードを短くしたりswitch文を使ったりするなど工夫が必要です。

では実際にやってみましょう。 `if-switch.js` を以下のように編集してください。

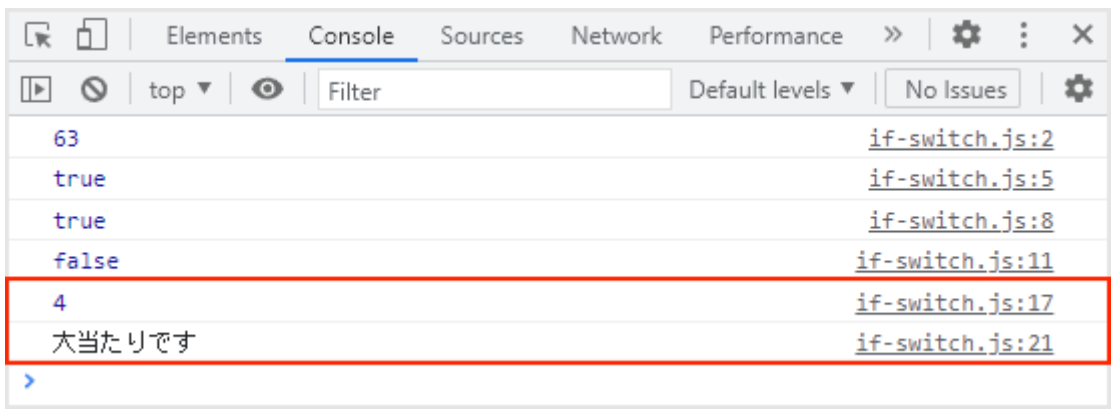
`if-switch.js`

```
//========= 前略 ========

// 変数numに0〜4までのランダムな整数を代入する
let num = Math.floor(Math.random() * 5);

// 変数numの値を出力する（確認用）
console.log(num);

// 変数numの値が4であれば、「大当たりです」という文字列を出力する
if (num === 4) {
  console.log('大当たりです');
}
+ // 変数numの値が3であれば、「当たりです」という文字列を出力する
+ else if (num === 3) {
+   console.log('当たりです');
+ }
// それ以外のときは、「はずれです」という文字列を出力する
else {
  console.log('はずれです');
}

```

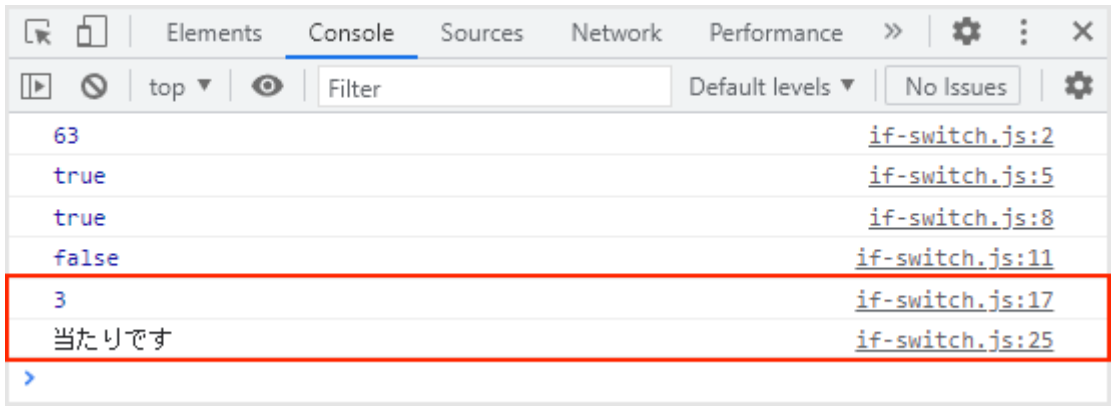
`index.html` をブラウザで開き、デベロッパーツールのコンソールを確認してみましょう。以下のように表示されればOKです。

- 変数 `num` の値が `4` のときは、「 `大当たりです` 」という文字列が表示される
- 変数 `num` の値が `3` のときは、「 `当たりです` 」という文字列が表示される
- それ以外のときは、「 `はずれです` 」という文字列が表示される

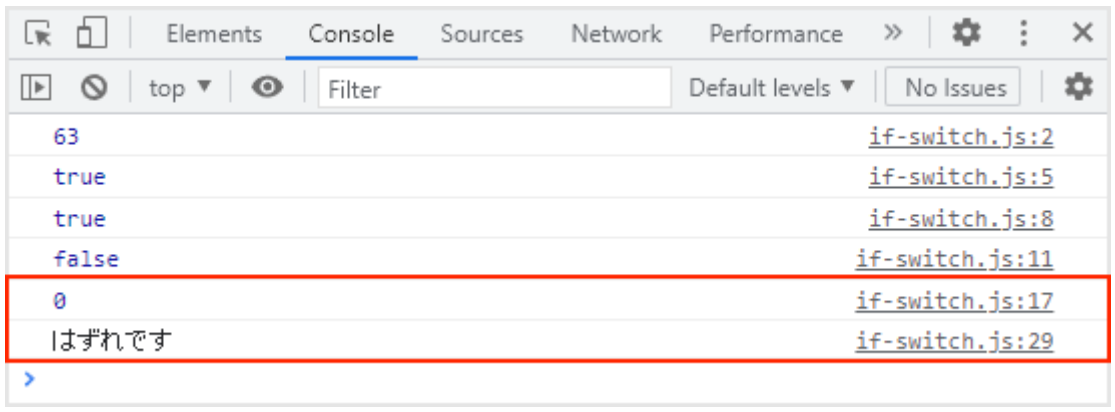
変数numの値が4のとき



変数numの値が3のとき



変数numの値がそれ以外（0、1、2）のとき



本章の学習は以上です。お疲れさまでした。

まとめ

本章では以下の内容を学習しました。

- 条件分岐とは、**条件によって処理を分けること**である
- 条件分岐を行うときは、**if文**または**switch文**を使う
- if文は「もし条件式が成り立つならば、処理を行う」という条件分岐である
- 条件が成り立つかどうかを判定する式のことを、**条件式**という
- 条件式には**比較演算子**を使う

```
if（条件式A）{
  条件Aが成り立つときの処理
}
else if（条件式B）{
  条件Bが成り立つときの処理
}
else {
  どの条件も成り立たないときの処理
}

```

比較演算子	処理の内容
==	2つの値が等しい場合は true を返す（等価演算子）。
===	2つの値とデータ型が等しい場合は true を返す（厳密等価演算子）。

比較演算子	処理の内容
!=	2つの値が等しくない場合は true を返す。
!==	2つの値とデータ型が等しくない場合は true を返す。
>	左辺の値が右辺の値よりも大きい場合は true を返す。
>=	左辺の値が右辺の値以上の場合は true を返す。
<	左辺の値が右辺の値よりも小さい場合は true を返す。
<=	左辺の値が右辺の値以下の場合は true を返す。

次章では、条件分岐のswitch文について学びます。

理解度を選択して次に進みましょう

ボタンを押していただくと次の章に進むことができます

～50%　50～80%　80～100%

最後に確認テストを行いましょう

下のボタンを押すとテストが始まります。

教材をみなおす　テストをはじめる

前に戻る　7 / 26 ページ　次に進む

一覧に戻る　改善点のご指摘、誤字脱字、その他ご要望はこちらからご連絡ください。