

🔗

🏠

🕒

📖

📄

🔍

✎

🔊

教材

🔍 検索

所属チーム ▼ 🔔<sup>1</sup> 👤

本文 目次 質問一覧 5件

ホーム 教材 JavaScriptの基礎を学ぼう オブジェクトを理解しよう

10章 オブジェクトを理解しよう

プログラミングにおける「オブジェクト」とは何かを解説します。

🕒120分 🏆 - 📖 読了

10.1 本章の目標

本章では以下を目標にして学習します。

▪ オブジェクトとは何か、概要をつかむこと

▪ オブジェクトの作り方、使い方を知ること

▪ オブジェクトを実際に使ってみること

前章では配列を使い、複数の同じようなデータをまとめて管理する方法を学びました。

では例えば、「名前」「年齢」「性別」「住所」「電話番号」など、異なる種類のデータをまとめて管理したい場合はどうでしょう  
か。

以下のように配列にすることも可能ですが、何番目に何のデータが入っているかがわかりにくいため、管理が難しくなってしまいま  
す。

1 // 配列

2 const personalData = [ '侍太郎', 36, '男性', '東京都', '020-0304-0506', '侍花子', 33, '女性', '京都府', '999-9999-9999' ]

3

そこで便利なのがオブジェクトです。オブジェクトを使えば以下のようにデータ名をつけられるので、異なる種類のデータも格段に管  
理しやすくなります（コードの書き方は本章で詳しく解説します）。

1 // オブジェクト

2 const personalData1 = { name: '侍太郎', age: 36, gender: '男性', address: '東京都', phoneNumber: '070-0809-1160' }

3 const personalData2 = { name: '侍花子', age: 33, gender: '女性', address: '京都府', phoneNumber: '999-9999-9999' }

4

本章でオブジェクトについて学び、異なる種類のデータもまとめて管理できるようになりましょう。

10.2 配列について復習しよう

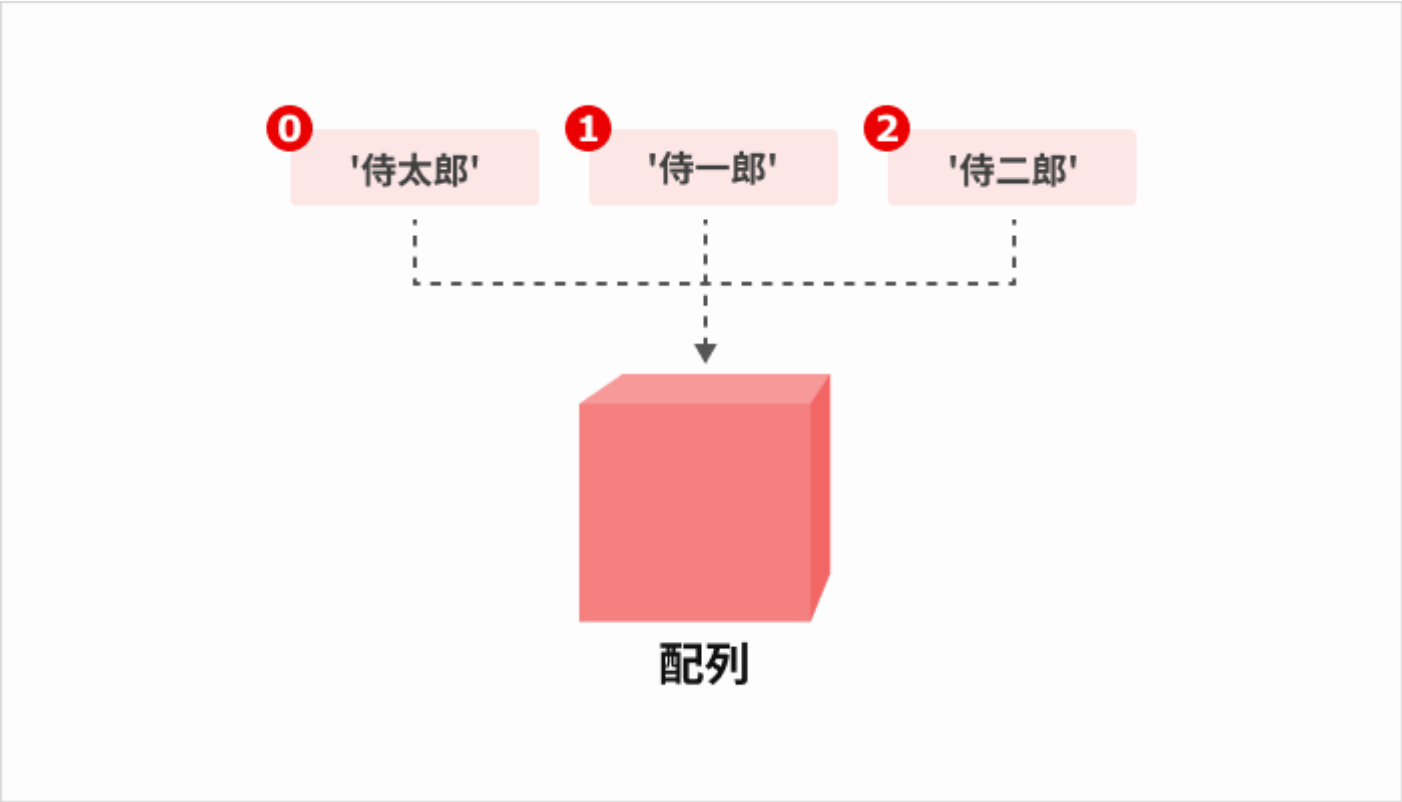
+ 質問する

https://terakoya.sejuku.net/programs/60/chapters/676

1/8

本章で学ぶオブジェクトは配列と性質が似ているので、まずは前章で学んだ配列について復習しておきましょう。

配列とは以下のように、**複数のデータのまとまり**のことでした。



また、配列の各要素には「0」から順番に番号（**インデックス**）が振られており、このインデックスを指定することで要素を取得・更新・追加することも学びました。

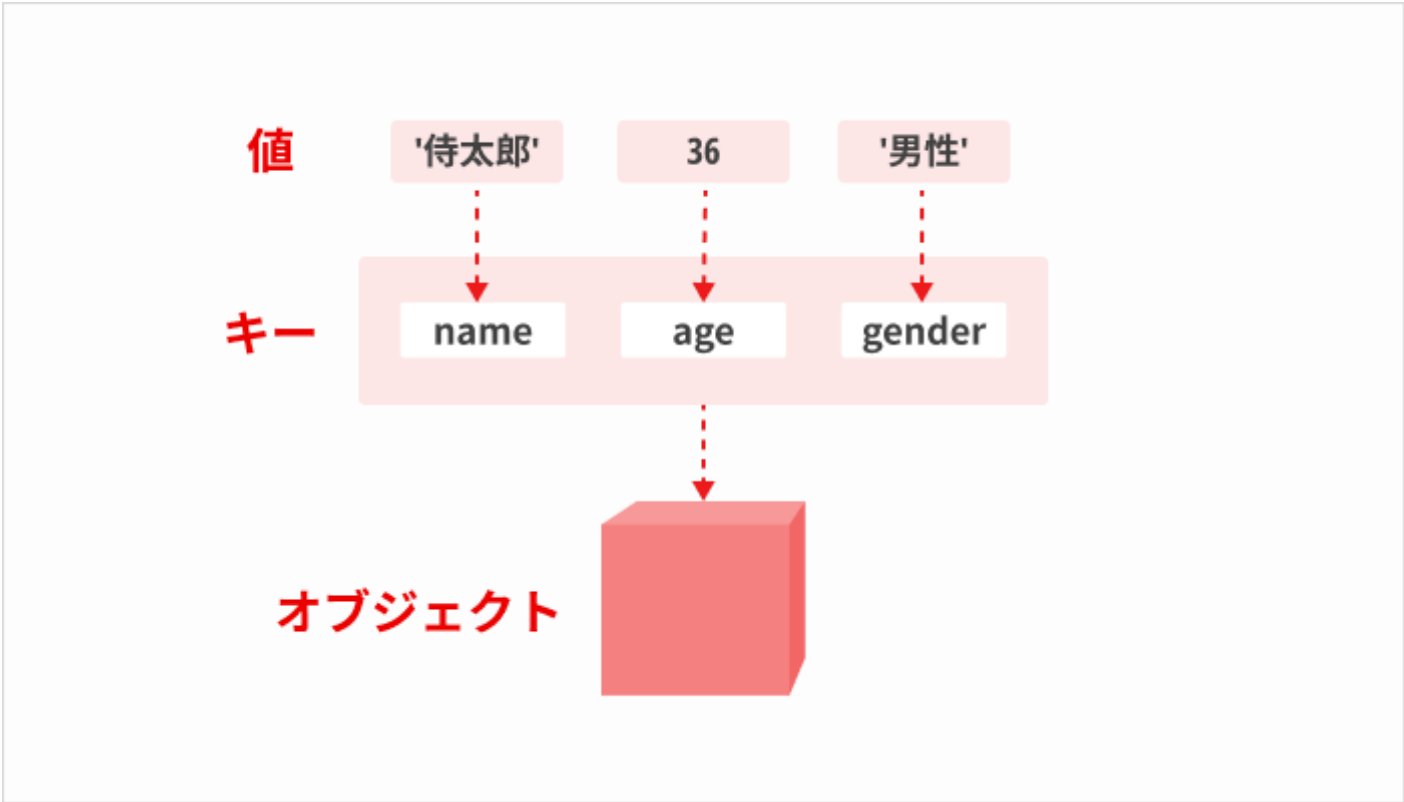
JSファイル（見本）

```
1  const userNames = ['侍太郎', '侍一郎', '侍二郎', '侍三郎', '侍四郎'];
2  // 2番目の要素を取得し、コンソールに出力する
3  console.log(userNames[1]);
4
5  // 2番目の要素を更新する
6  userNames[1] = '侍花子';
7
8  // 6番目に要素を追加する
9  userNames[5] = '侍五郎';
10
```

### 10.3 オブジェクトとは

オブジェクトとは、配列におけるインデックスの代わりに**キー**と呼ばれるラベルをつけて管理するデータのまとまりのことです。

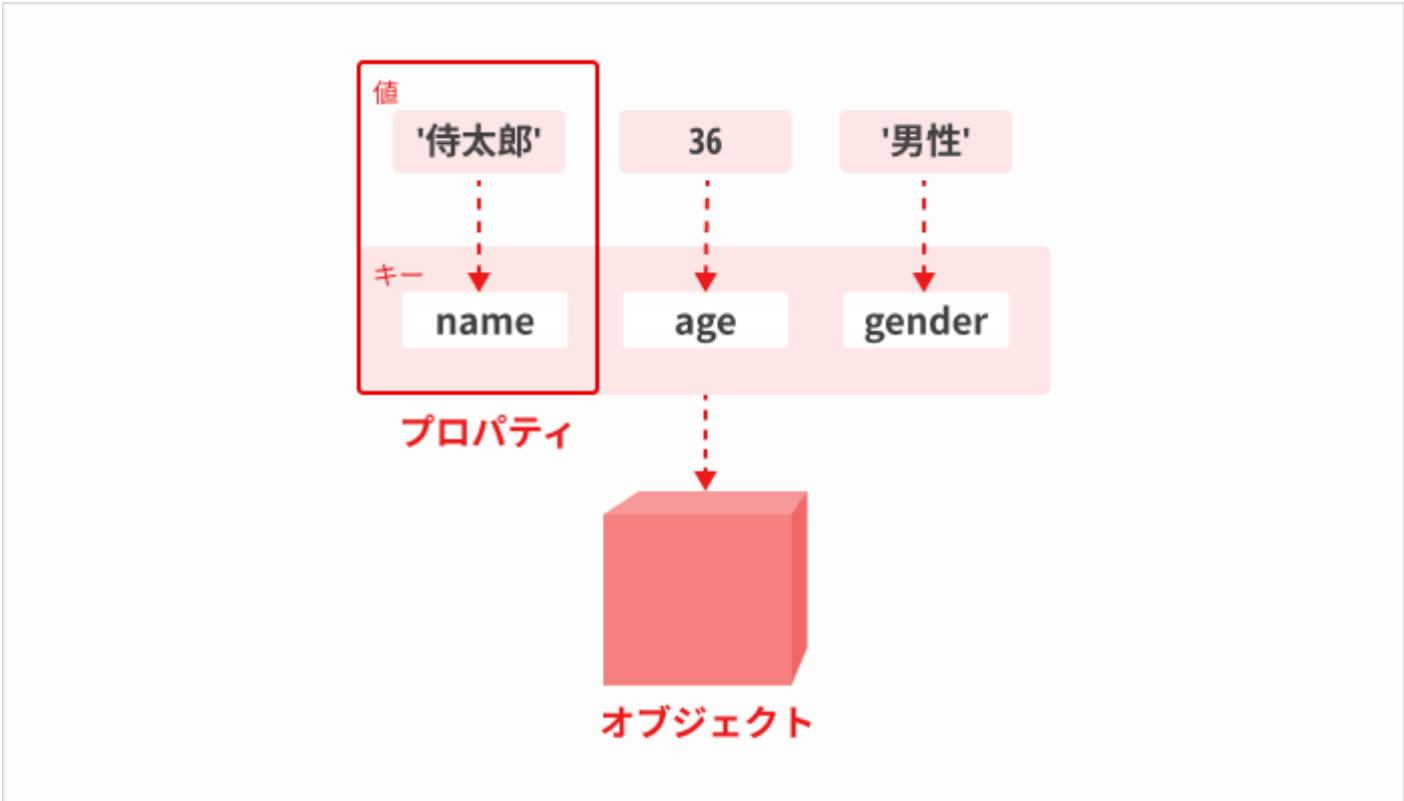
- キー = 値が何を表すのかわかりやすく名前をつけたもの



「複数のデータのまとまり」という点では配列と性質が似ており、他のプログラミング言語では**連想配列**などと呼ばれることもあります。

- 配列：各要素に「0」から順番に番号（**インデックス**）を振って管理する
- オブジェクト：インデックスの代わりに**キー**と呼ばれるラベルをつけて管理する

なお、オブジェクトにおけるキーと値のセットのことを、**プロパティ**といいます。配列と同じように、要素と呼ばれることもあります。



## 10.4 オブジェクトの作り方・使い方

オブジェクトを作るには、以下のように { キー1: 値1, キー2: 値2, キー3: 値3, ..... } と書きます。

JSファイル（見本）

```
1 const personalData = { name: '侍太郎', age: 36, gender: '男性' };
2
```



このように、オブジェクトは `{ }` の中にカンマ区切りでプロパティ（キーと値のセット）を入れていきます。また、キーと値の間にはコロン `:` を記述します。

## 値を取り出す方法

オブジェクトの値を取り出す方法は、ブラケット記法とドット記法の2通りあります。

- 1. ブラケット記法： オブジェクト名[ 'キー名' ] （例： `personalData[ 'gender' ]` ）
- 2. ドット記法： オブジェクト名. キー名 （例： `personalData.gender` ）

ブラケット記法は配列と似たような方法です。 オブジェクト名[ 'キー名' ] のように記述します（ブラケット＝角括弧 `[ ]` ）。なお、値を取り出すときはキー名をシングルクォーテーションまたはダブルクォーテーションで囲む必要がある点に注意しましょう。

例えば以下のオブジェクト `personalData` から性別を取得したいのであれば、 `personalData[ 'gender' ]` と記述します。

JSファイル（見本）

```
1 const personalData = { name: '侍太郎', age: 36, gender: '男性' };
2
3 // 'gender' というキーを持つ値（'男性'）が出力される
4 console.log(personalData[ 'gender' ]);
5
```

続いて2つ目のドット記法は、 オブジェクト名. キー名 のように記述する方法です。ブラケット記法と全く同じ結果になります。

JSファイル（見本）

```
1 const personalData = { name: '侍太郎', age: 36, gender: '男性' };
2
3 // 以下のいずれも同じ値が出力される
4 console.log(personalData[ 'gender' ]);
5 console.log(personalData.gender);
6
```

どちらの方法でも構いませんが、一般的にはドット記法のほうが多く使われています。よって、本教材でもドット記法で記述します。

## 値を更新・追加する方法

オブジェクトの値を**更新**したり、**追加**したりすることもできます。その際も以下のように、キーを使います。

JSファイル（見本）

```
1 const personalData = { name: '侍太郎', age: 36, gender: '男性' };
2
3 // 'age' というキーの値を更新する
4 personalData.age = 37;
5
6 // 新しくプロパティ（キーと値）を追加する
7 personalData.address = '東京都';
8
```



なお、新しいプロパティはオブジェクトの末尾に追加されます（このあと5節で実際に確認します）。

## 10.5 オブジェクトを使ってみよう

では、実際にオブジェクトを使ってみましょう。まずはVisual Studio Codeを開き、 `js` フォルダ内に新しく `object.js` というファイルを作成してください。

続いて、 `object.js` を以下のように編集しましょう。オブジェクト `personalData` を宣言・定義したあとに、値を更新・追加します。

object.js

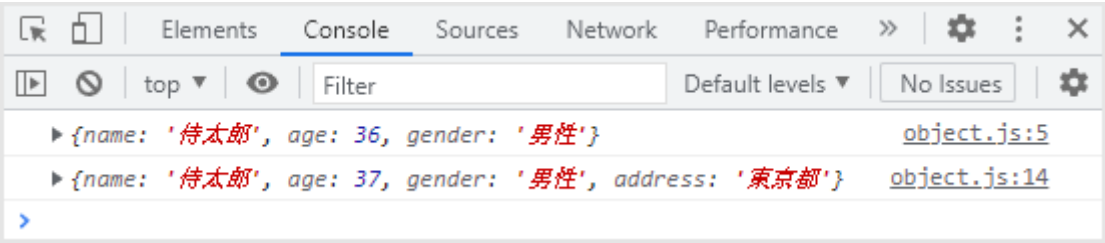
```
1 + // オブジェクトの宣言と値の代入を行う
2 + const personalData = { name: '侍太郎', age: 36, gender: '男性' };
3 +
4 + // オブジェクトの値を出力する
5 + console.log(personalData);
6 +
7 + // 'age'というキーの値を更新する
8 + personalData.age = 37;
9 +
10 + // 新しくプロパティ（キーと値）を追加する
11 + personalData.address = '東京都';
12 +
13 + // オブジェクトの値を出力する
14 + console.log(personalData);
15
```

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

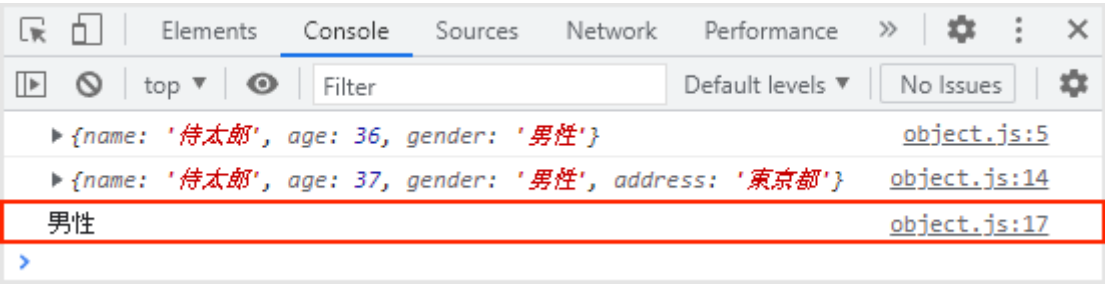
index.html

```
1 <!DOCTYPE html>
2 <html lang="ja">
3
4 <head>
5   <meta charset="UTF-8">
6   <title>JavaScript基礎編</title>
7 </head>
8
9 <body>
10 -   <script src="js/array.js"></script>
11 +   <script src="js/object.js"></script>
12 </body>
13
14 </html>
15
```

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、値を更新・追加する前後でオブジェクトの中身が変わっていればOKです。



最後に演習として、オブジェクト `personalData` の `'gender'` というキーの値をコンソールに出力してみてください。以下のようにコンソールに表示されればOKです。



正解のコードは以下のとおりです。

object.js

```
1 // オブジェクトの宣言と値の代入を行う
2 const personalData = { name: '侍太郎', age: 36, gender: '男性' };
3
4 // オブジェクトの値を出力する
5 console.log(personalData);
6
7 // 'age'というキーの値を更新する
8 personalData.age = 37;
9
10 // 新しくプロパティ（キーと値）を追加する
11 personalData.address = '東京都';
12
13 // オブジェクトの値を出力する
14 console.log(personalData);
15
16 + // 'gender'というキーの値を出力する
17 + console.log(personalData.gender);
18
```

## 補足：オブジェクトの宣言にconstを使う理由

本章では、オブジェクトを宣言するときに `const` を使いました。これは配列と同じ理由で、`const` で定数を宣言したとしても、以下のようにオブジェクトの要素を更新したり、追加したりすることはできるからです。

JSファイル（見本）

```
1 // オブジェクトの宣言と値の代入を行う
2 const personalData = { name: '侍太郎', age: 36, gender: '男性' };
3
4 // 'age'というキーの値を更新する
5 personalData.age = 37;
6
7 // 新しくプロパティ（キーと値）を追加する
8 personalData.address = '東京都';
9
```

ただし、以下のようにオブジェクトの値を再代入する（丸ごと入れ替える）とエラーが発生します。

JSファイル（見本）



```
1 // オブジェクトの宣言と値の代入を行う
2 const personalData = { name: '侍太郎', age: 36, gender: '男性' };
3
4 // 値を再代入する（丸ごと入れ替える）とエラーが発生する
5 personalData = { name: '侍花子', age: 33, gender: '女性' };
6
```

しかし、このように一度宣言したオブジェクトに値を再代入する（丸ごと入れ替える）ケースはほとんどないため、オブジェクトも配列と同じように `const` を使うのが一般的です。

最後に、もう一度 `let` と `const` の違いを復習しておきましょう。

- `let` : 変数を宣言するときに使う。あとから中身を入れ替えられる（再代入できる）
- `const` : 定数を宣言するときに使う。あとから中身を入れ替えられない（再代入できない）

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。

- オブジェクトとは、配列におけるインデックスの代わりに**キー**と呼ばれるラベルをつけて管理するデータのまとまりのことである
- オブジェクトにおけるキーと値のセットのことを、**プロパティ**という
- オブジェクトは { キー1: 値1, キー2: 値2, キー3: 値3, ..... } のように、 { } の中にカンマ区切りでプロパティを入れていく
- オブジェクトの値を取得・更新・追加する方法
  - 1. ブラケット記法： オブジェクト名[ 'キー名' ] （例： `personalData[ 'gender' ]` ）
  - 2. ドット記法： オブジェクト名. キー名 （例： `personalData.gender` ）


次章では、関数について学びます。

### 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

13 / 26 ページ

次に進む

く 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。

© SAMURAI Inc.    [利用規約](#)   [法人会員利用規約](#)   [プライバシーポリシー](#)   [運営会社](#)