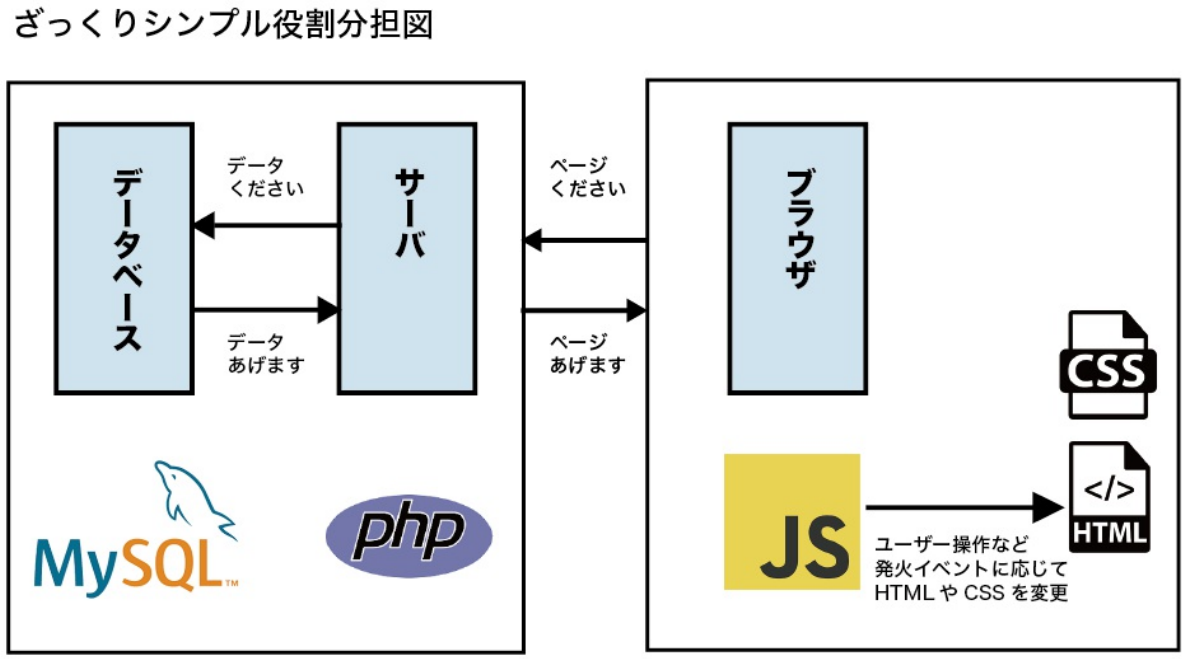
Vue.js勉強会 vol.1

Hello Worldから始めるVue.js

対象者

Vue.jsというかJavaScript初心者。
デザイナーも含めるので、基礎の基礎からやってみます。

JavaScriptってなに



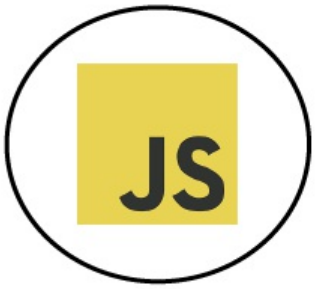
PHP と JavaScript の ちがい

項目	JavaScript	PHP
実行環境	ブラウザで動くので、ファイルを作成するだけでOK。	サーバーが必要。ファイルだけでは動かない。
実行 タイミング	必ずイベントを指定する。ページロード時・クリック時、などページ表示のタイミングのみ。	ページ表示のタイミングのみ。
HTMLの 変更部分	ページ全部を書き換えなくてもいい。変えたい部分を変更するだけでOK(非同期)	1箇所を書き換えるだけでも、１ページ丸々再度描画しないといけない。(同期)

ページ全部を変えたい時はPHP。
ちょっとだけ変えたい時はJavaScript。

何が違うんですか？

- JavaScript
- jQuery
- Vue.js



言語

フレームワーク

ライブラリ

はぁ？？？？（ ゜Д゜）

↑予測できる愁さんの顔

ライブラリとフレームワークのちがい

jQueryっていうライブラリ

よく使うものを集めたものがライブラリ。パーツ集です。

jQureyは、JavaScriptをより扱いやすく、簡単に書けるようにしたもの。

そのライブラリの決まった記述で呼び出して扱う。

[参考コード] https://jsfiddle.net/pocchi/udhacmdo/8/

簡単なhoverの処理ですが、
jQueryで4行、JavaScriptで8行になってます。すっきり。

ライブラリとフレームワークのちがい

Vue.jsというかフレームワーク

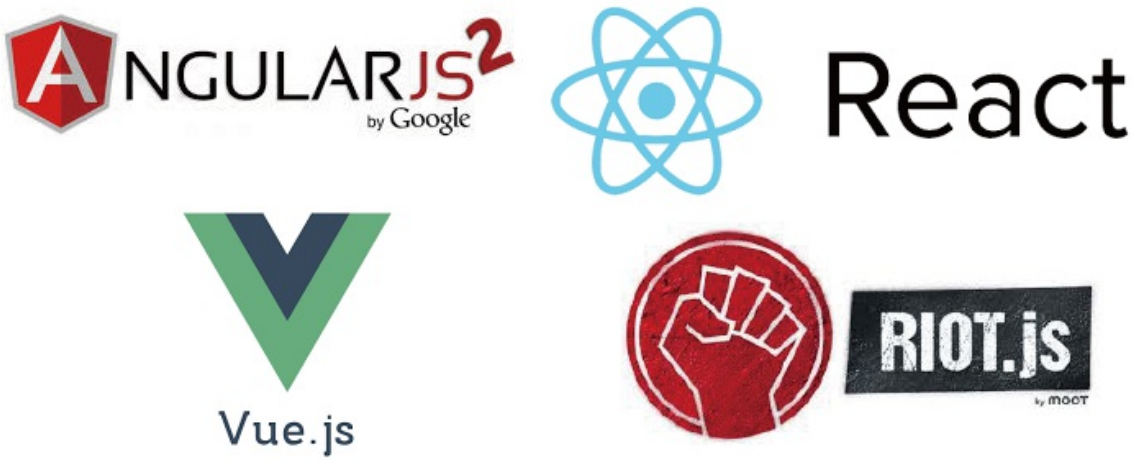
すでに決まっている設計に基づき、開発者は基本的には、そのひな形に沿ってコードを書いて行く。

[参考コード] https://jsfiddle.net/chrisvfritz/50wL7mdz/

ライブラリが部品なら、フレームワークはひな形
なので、一緒に使うことも可能。

Vue.jsと他のJSフレームワーク

他にももちろんいっぱいJSフレームワークはあります。



などなどなど・・・いっぱいあるので、
開発に合わせて選択すること、
流行り廃れをなんとなく知っておくが必要。

Vue.jsの良きところ

- 学習コストが比較的低めで、自分の習熟度に合わせて取り入れられる
- 小中規模のシステムに向いている
- 他フレームワークの良いところをあわせもっている
- 日本語リファレンスが充実していてわかりやすい。

・・・なので、JSフレームワークこと初めには良いんじゃないでしょうか。

では、はじめてみましょう！

Vue.jsの始め方(初心者用)

htmlを作成し、CDN読み込んじゃう

```
<!-- Vue.jsの読み込み -->
<script src="https://unpkg.com/vue"></script>
```

jsFiddleってやつ

https://jsfiddle.net/chrisvfritz/50wL7mdz/
今回はこれ使います。
Forkを押して、自由に書き換えましょう
Runで実行されます。保存はUpdate。
保存しなくてもRunできます。
URLを覚えておけば、いつでもアクセスできます。

---

messageを変更してみよう

```
new Vue({
  el: '#app',
  data: {
    message: 'ここを変更してみて'
  }
})
```

変更後、Runをクリック！

[こうなってるはず] https://jsfiddle.net/pocchi/rtec7eyh/1/

こんな書き方もできる

```
<script src="https://unpkg.com/vue/dist/vue.js"></script>

<div id="app">
  <p v-html="message">メッセージ</p>
</div>
```

v-htmlという属性をつければはじめに書いておいて、
置き換えることもできる。
{{生の文字列}}だが、v-htmlはHTMLとして扱える。

[こうなってるはず] https://jsfiddle.net/pocchi/rtec7eyh/2/

---

v-ifで出し分け

```
<div id="app">
  <p v-if="message">メッセージ</p>
</div>
```

```
new Vue({
  el: '#app',
  data: {
    message: true
  }
})
```

[こうなっているはず] https://jsfiddle.net/pocchi/rtec7eyh/3/

`message: false`に変更してみましょう

[変更後はこれ] https://jsfiddle.net/pocchi/rtec7eyh/4/

ところでv-ってなに

v- から始まる特別な属性をディレクティブと言います。

単一の JavaScript 式を期待します。
ディレクティブの仕事は、属性値の式が変化したときに、リアクティブに副作用を ※DOM に適用すること。

さっきの例だと、
`true`で挿入、`false`で削除できるってこと

※DOM（Document Object Model）とは、xmlやhtmlの各要素、たとえば<p>とか<img>とかそういった類の要素にアクセスする仕組みのことです。このDOMを操作することによって、要素の値をダイレクトに操作できます。

--- ---

Vue.js側で値を変化させれば、
それにつられてHTMLも勝手に変わる

つまり、状態を管理しやすい ！

---

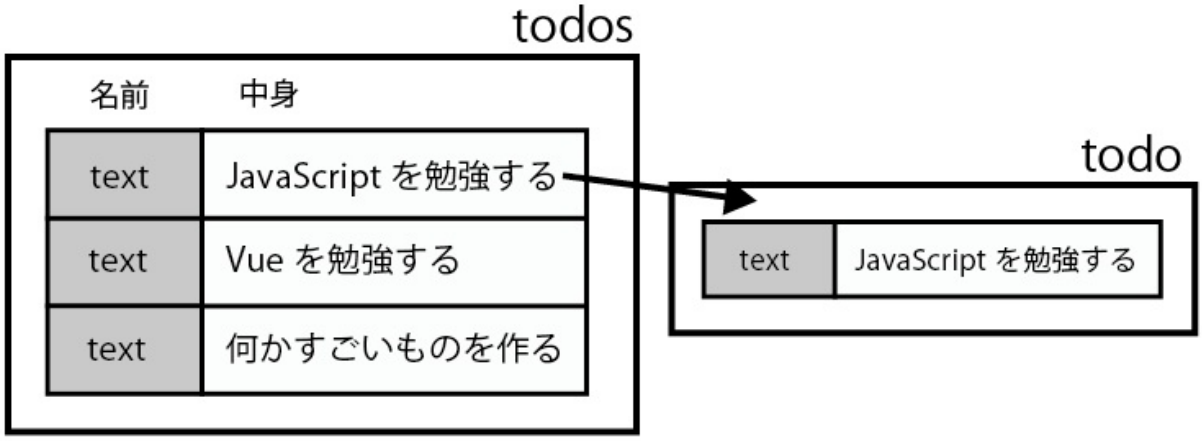
v-forでリストを表示する

```
<div id="app">
  <ol>
    <li v-for="todo in todos">
      {{todo.text}}
    </li>
  </ol>
</div>
```

```
new Vue({
  el: '#app',
  data: {
     todos: [
      { text: 'JavaScriptを勉強する' },
      { text: 'Vueを勉強する' },
      { text: '何かすごいものを作る' }
    ]
  }
})
```

[こうなっているはず] https://jsfiddle.net/pocchi/rtec7eyh/5/

何をやっているのか



todosの中身をtodoの中に上から1つずつ移し替えている

TODOリストを作ってみましょう！

TODOリストのTODOリスト

- やることを入力する
- やることリストを見る
- やることが終わったら消す

TODOリストのTODOリスト

- やることを入力する
- やることリストを見る-->すでにできている！
- やることが終わったら消す

やることを入力できるようにしよう

```
<div id="app"><!-- この下から追記 -->
  <input v-model="inputText" type="text" />
  {{inputText}}<!-- この上まで追記 -->
  <ol>
    <li v-for="todo in todos">
      {{todo.text}}
    </li>
  </ol>
</div>
```

```
new Vue({
  el: '#app',
  data: {/* この下から追記 */
     inputText: null,
     todos: [/* この上まで追記 */
      { text: 'JavaScriptを勉強する' },
      { text: 'Vueを勉強する' },
      { text: '何かすごいものを作る' }
//以下省略
```

[こうなっているはず]https://jsfiddle.net/pocchi/3LnLuc4p/1/

フォームに入力すると値が取得できることは確認できた！

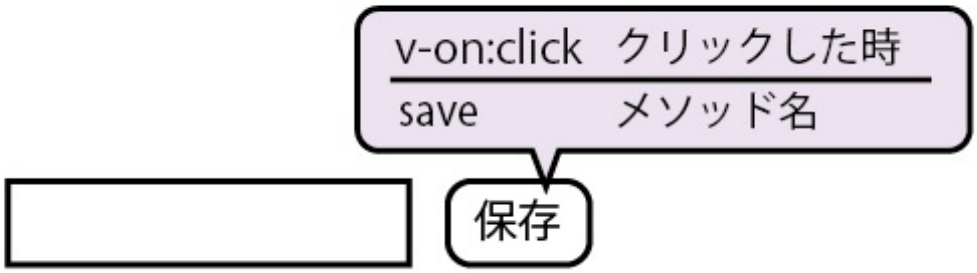
次は保存ボタンをつけたい！

```
<div id="app">
  <input v-model="inputText" type="text" />
  <input type="submit" value="保存" v-on:click="save"/>
  <ol>
    <li v-for="todo in todos">
      {{todo.text}}
    </li>
  </ol>
</div>
```

```
new Vue({
  el: '#app',
  data: {
     /* --inputTextとtodosは中略-- */
  },/* この下から追記 */
  methods:{
    save: function(){
      this.todos.push({text: this.inputText});
    }
  }
})/* この上まで追記 */
```

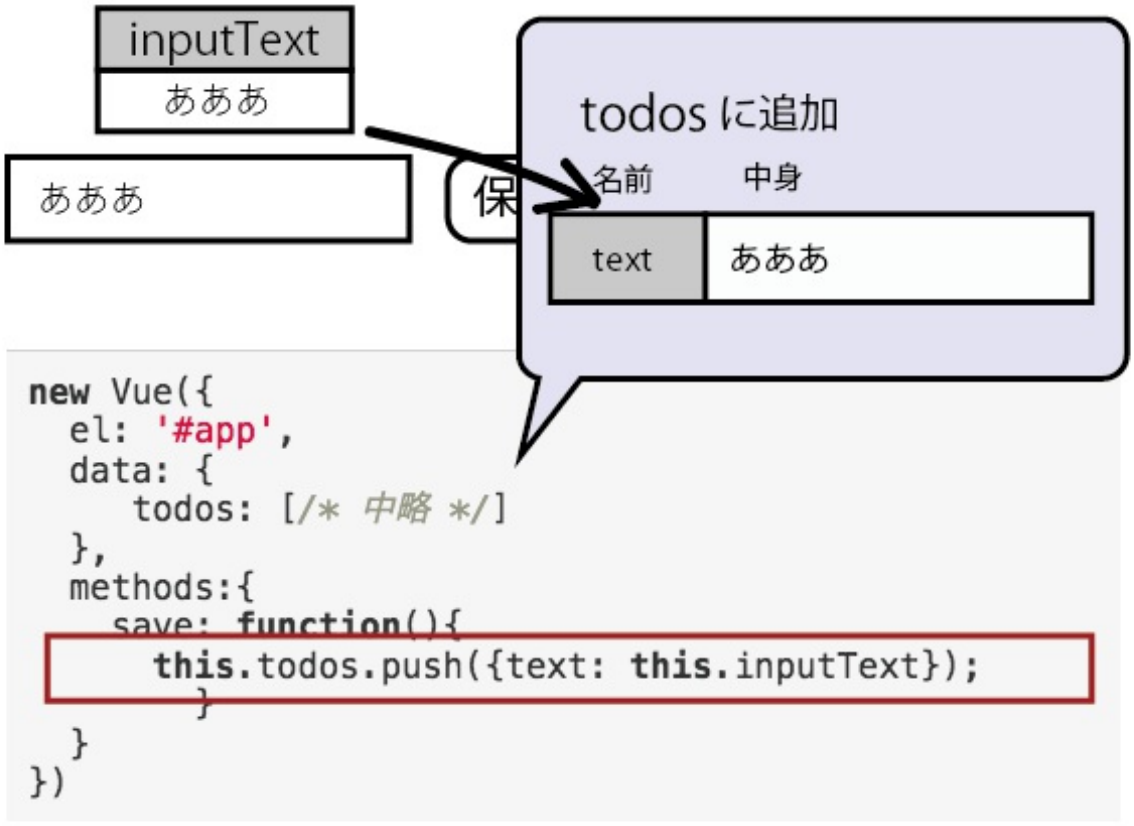
[こうなっているはず]https://jsfiddle.net/pocchi/3LnLuc4p/6/

どうなっているのか




```
new Vue({
  el: '#app',
  data: {
     todos: [/* 中略 */]
  },
  methods:{
    save: function(){
      this.todos.push({text: this.inputText});
        }
  }
})
```

どうなっているのか



これで、保存ができる！

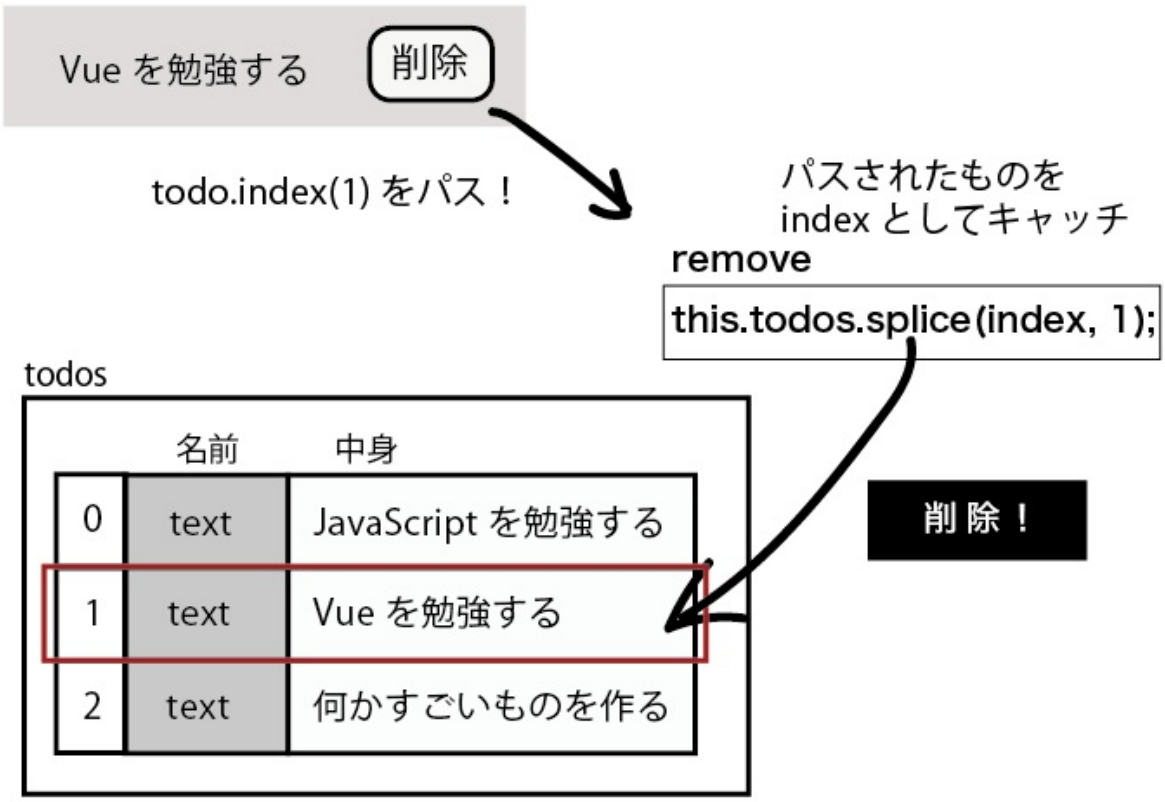
後は削除するにはどうしたらいいのか！

```
<div id="app">
  <input v-model="inputText" type="text" />
  <input type="submit" value="保存" v-on:click="save"/>
  <ol>
    <li v-for="todo in todos">
      {{todo.text}}<!-- この下から変更！ -->
      <input type="submit"
      value="削除"
      v-on:click="remove(todo.index)" />
    </li><!-- この上まで変更！ -->
  </ol>
</div>
```

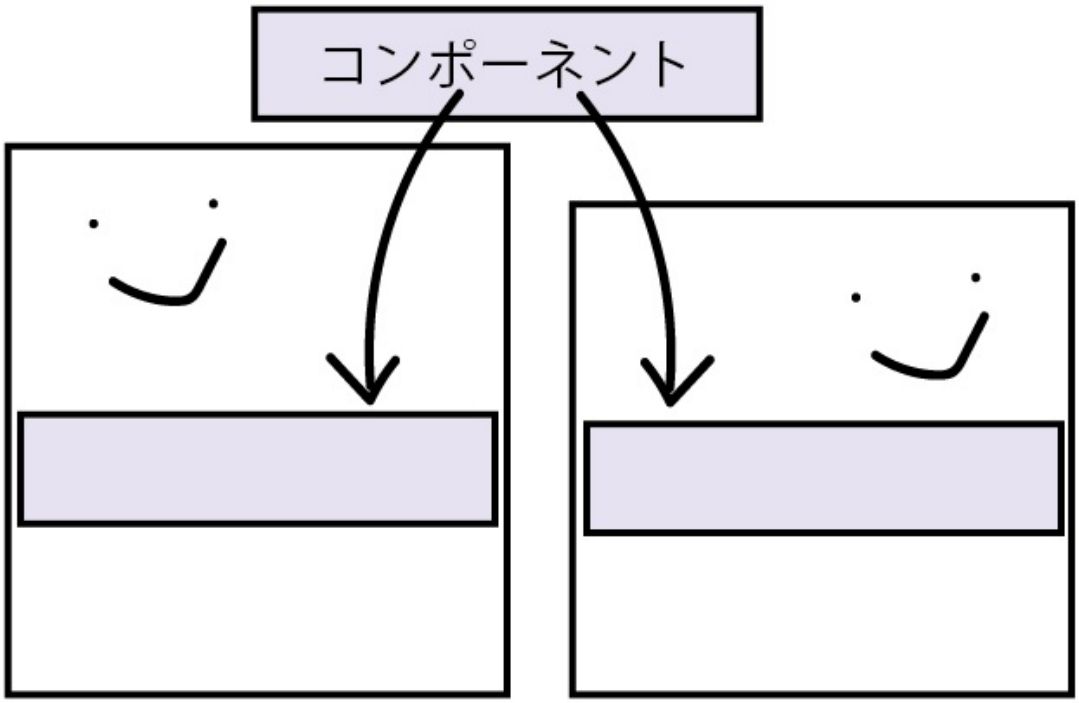
```
methods:{
   save: function(){
     this.todos.push({text: this.inputText});
   },/* <-コンマをつけて、この下から追記 */
   remove: function(index){
       this.todos.splice(index, 1);
   } /* ここまで追記 */
 }
```

[こうなっているはず]https://jsfiddle.net/pocchi/3LnLuc4p/10/

何が起こっている？



コンポーネント化してみましょう！



Sketchでいうシンボルに近い

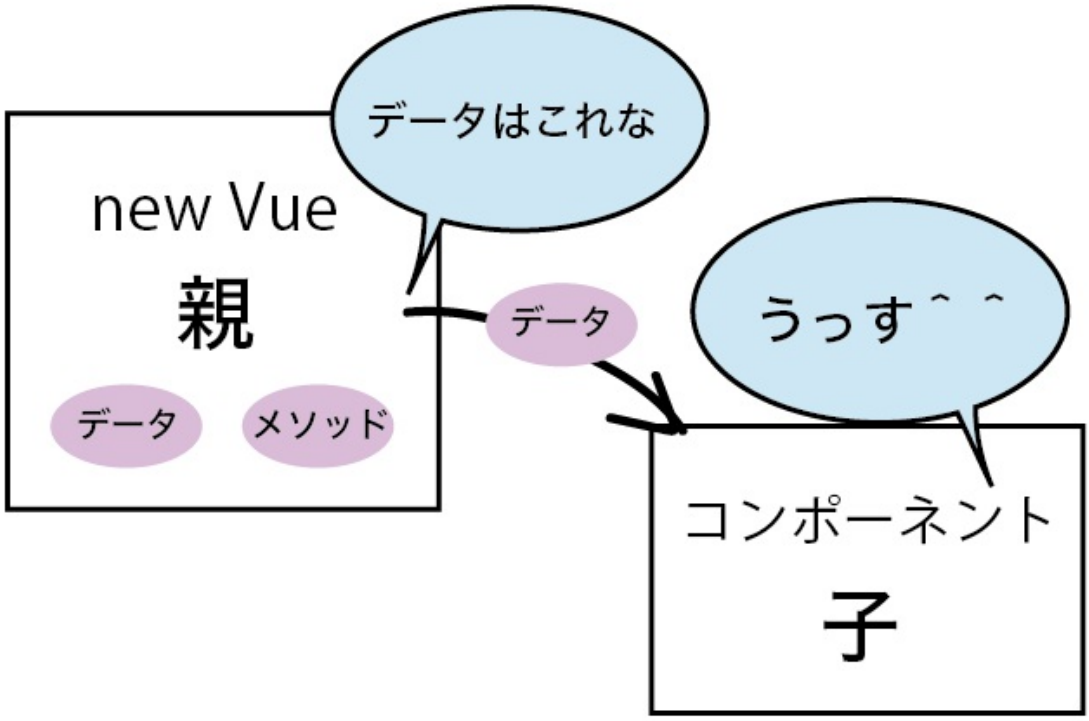
```
<div id="app">
  <!-- 中略 -->
  <ol>
      <list-component
      v-for="todo in todos"
      v-bind:key="todo"
      v-bind:text="todo.text"
      v-bind:text="todo.index"
      ></list-component>
  </ol>
</div>
```

new Vue({の上に追加

```
Vue.component('list-component', {
  template: '<li>{{text}}
              <input
               type="submit"
               value="削除"
               v-on:click="remove(index)" />
              </li>',
  props:['text','index']
});
```

[こうなっているはず]https://jsfiddle.net/pocchi/3LnLuc4p/12/

削除が動かない！



メソッドも渡してあげよう！

```
<list-component
      v-for="todo in todos"
      v-bind:key="todo"
      v-bind:text="todo.text"
      v-bind:index="todo.index"
      v-on:remove="remove(todo.index)"
      ></list-component>
```

```
Vue.component('list-component', {
  template: `<li>{{text}}
             <input type="submit"
             value="削除"
             v-on:click="$emit('remove')"/></li>`,
  props:['text','index']
});
```

template: `` で囲まれていることに注意！
[こうなっているはず]https://jsfiddle.net/pocchi/3LnLuc4p/15/

動きましたか？

おわり！

参考

公式(日本語)

https://jp.vuejs.org/
