

Vue Data Binding

Agenda

Obsługa zdarzeń

Wiązanie dwukierunkowe

Przechowywanie danych lokalnie

Handle Events

HTML

```
<p onClick="myFunc()">  
  some text  
</p>
```

VUE

```
<p v-on:click=  
  "name='Hello'">  
  {{name}}  
</p>
```

Event Handler Object

Obiekt `$event`

Właściwości

`target` – **obiekt DOM (element) wyzwalający zdarzenie**

`type` – **rodzaj zdarzenia**

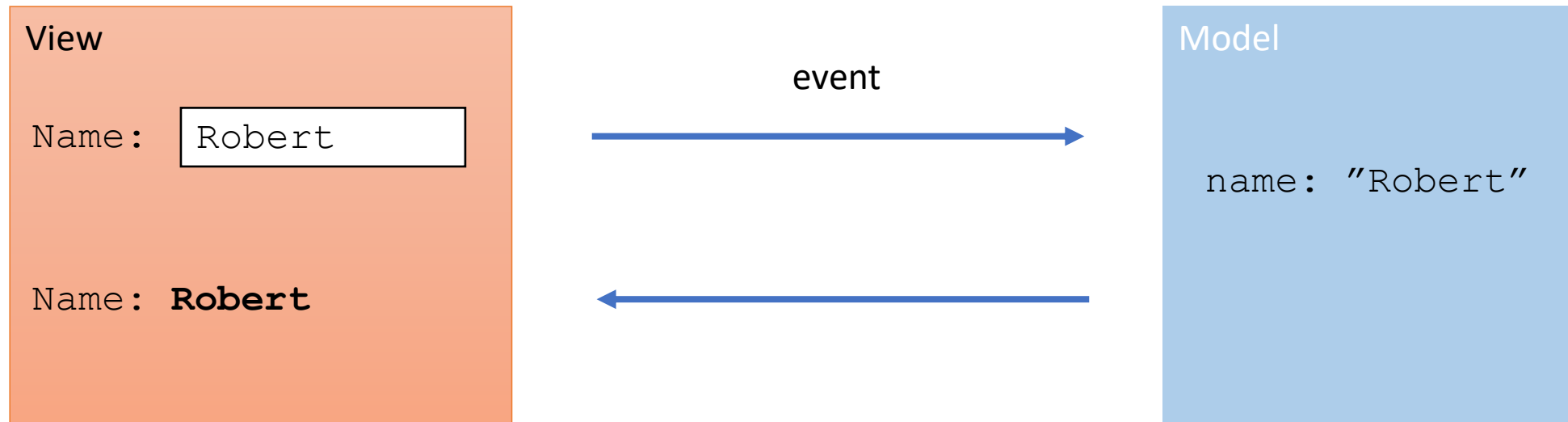
`key` – **naciśnięty klawisz**

Vue Two Way Data Binding

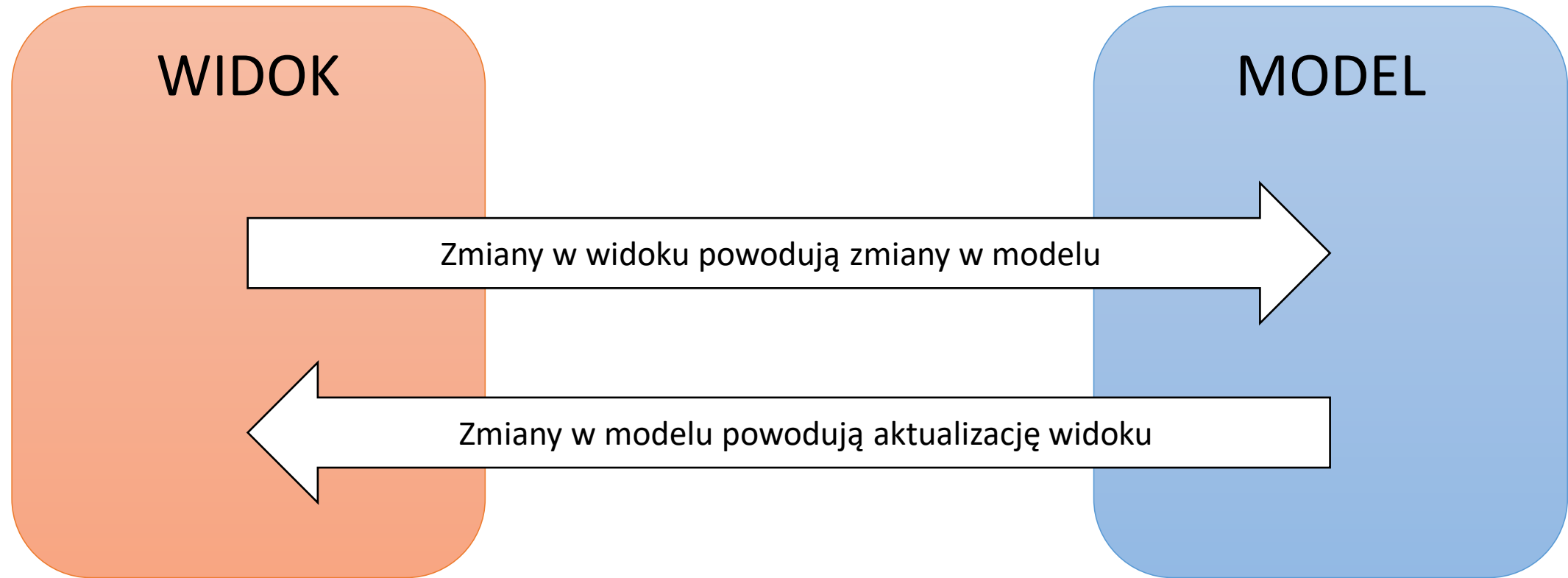
```
<p v-on:click="changeName">{{ name }}</p>
```

```
<script>
  var app = new Vue({
    data: {
      name: "Adam"
    },
    methods: {
      changeName: function ($event) {
        this.name = "Barbara";
      }
    }
  })
</script>
```

Two Way Data Binding



View-Model Two Way Data Binding



Form Input Binding

Input

pole tekstowe

pole opcji

pole wyboru

Select

Textarea

V-model directive

Name	Value
Name	<input type="text"/>
Sex	<input type="radio"/> Male <input checked="" type="radio"/> Female
Eye color	<input type="text" value="green"/>
Check all that apply	<input type="checkbox"/> Over 6 feet tall <input type="checkbox"/> Over 200 pounds
Describe your athletic ability: <input type="text"/>	
<input type="button" value="Enter my information"/>	

Binding to Text Fields

```
<input type="text" v-model="property">
```

```
<textarea v-model="property"></textarea>
```

V-model Modifiers

`number` – zamiana łańcucha znaków na liczbę

`trim` – usuwanie białych znaków

`lazy` – modyfikacja właściwości dopiero po utracie focusa

```
<input v-model.trim="studentName">  
<input v-model.number="studentAge" type="number">
```

```
// model  
data: {  
  studentName: "",  
  studentAge: 0  
}
```

Binding to Checkbox

```
<input type="checkbox"  
      id= "student"  
      v-model="isStudent">
```

```
<label for="student">Student UEK</label>
```

Binding to Multiple Checkboxes

```
<input type="checkbox" id="ser"
value="ser" v-model="dodatk1">
<label for="ser">Ser</label>
```

```
<input type="checkbox" id="keczup"
value="keczup" v-model="dodatk1">
<label for="keczup">Keczup</label>
```

```
data: {
  dodatki: []
}
```

Binding to Radio

```
<input type="radio" id="normalny"  
value="normalny" v-model="bilet">  
<label for="normalny">Normalny</label>
```

```
<input type="radio" id="ulgowy"  
value="ulgowy" v-model="bilet">  
<label for="ulgowy">Ulgowy</label>
```

```
data: {  
  bilet: ''  
}
```

Binding to Select

```
<select v-model="pizza">
  <option disabled value="">Wybierz:</option>
  <option>margherita</option>
  <option>siciliana</option>
  <option>neapolitana</option>
</select>

data: {
  pizza: ''
}
```

Local Data Storage

Web Storage API

Local / Session Storage objects

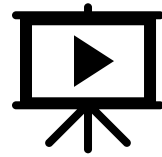
window.localStorage | window.sessionStorage

Storing key / value pairs, string values only

Methods: setItem / getItem for data storing / retrieving

Tutorials: [w3schools](#) [MDN](#)

Local Storage Examples



<https://www.youtube.com/watch?v=ZYmYysUGjbQ>

To do

Data Binding

Program first-input.html umożliwia wprowadzenie imienia osoby. W polu formularza wprowadź swoje imię, a następnie:

Korzystając z konsoli, wyświetl imię na konsoli

Korzystając z konsoli, zmień imię na inne oraz sprawdź, czy zostało zmienione w polu formularza

Korzystając z Vue Devtools, wyświetl wprowadzone imię

Korzystając z Vue Devtools, zmień imię na inne oraz sprawdź, czy zostało zmienione w polu formularza

Book

Utwórz program book.html, zawierający pola formularza do wprowadzenia tytułu książki oraz imienia i nazwiska jej autora. Po kliknięciu na przycisk "Zapisz", program powinien wyświetlić wprowadzone dane w konsoli.

Aby wyświetlić dane na konsoli, w aplikacji Vue utwórz funkcję, która wywołana po wystąpieniu zdarzenia, wyświetli wymagane dane.

Counter

Utwórz program counter.html wyświetlający przycisk wraz z informacją o liczbie jego kliknięć. Liczbę kliknięć wyświetl na przycisku. Kliknięcie przycisku powinno skutkować zwiększeniem licznika kliknięć o 1.

BMI

Program bmi.html umożliwia wprowadzenie wzrostu i masy ciała. Uzupełnij program, aby obliczał i wyświetlał wskaźnik masy ciała BMI.

Dla obliczenia oraz wyświetlenia wskaźnika masy ciała zastosuj właściwość 'computed'.

Elective Courses

Utwórz aplikację electivecourses.html, w którym wyświetl nazwy pięciu przedmiotów do wyboru z programu studiów, wraz z polami wyboru (checkbox). Aplikacja powinna wyświetlać informację, ile przedmiotów zostało wybranych.

Aby uzyskać informację o liczbie wybranych przedmiotów, odczytaj liczbę elementów tablicy.

ToDo List

Aplikacja todo.html umożliwia dodanie czynności do wykonania.

Korzystając z Vue Devtools sprawdź, czy dodawane czynności zapisywane są w modelu.

Zmodyfikuj aplikację, aby dodawane czynności wyświetlane były w postaci listy numerowanej.

CV zawiera następujące dane:

imię, nazwisko, wiek (wartość całkowita), płeć (kobieta, mężczyzna – pole opcji), wykształcenie (brak, podstawowe, średnie, wyższe – lista wyboru), umiejętności (obsługa komputera, prawo jazdy, kreatywność, praca w zespole).

Utwórz formularz umożliwiający wprowadzenie danych zawartych w CV. Wprowadzone dane zapisz we właściwościach obiektu.

Wprowadź dane oraz sprawdź w konsoli oraz w Vue Devtools, czy wprowadzone dane zostały zapisane w modelu.

CV in Local Storage

Zmodyfikuj program cv.html, aby możliwe było trwałe przechowanie danych w Local Storage. Dodaj przycisk Zapisz, którego kliknięcie zapisze dane do Local Storage. Dodaj przycisk Czytaj, aby odczytać dane zapisane w Local Storage.

Skonwertuj obiekt JS do tekstu i zapisz w local storage.

Skonwertuj tekst odczytany z Local Storage do obiektu JS.

Sprawdź w Chrome Devtools, czy dane zostały zapisane w Local Storage.

Deleted Items

Zmodyfikuj aplikację todo.html, aby podwójne kliknięcie na dodaną wcześniej czynność powodowało usunięcie jej z listy.