# Reaktywne aplikacje frontendowe

Projektowanie i programowanie systemów internetowych II

mgr inż. Krzysztof Rewak

14 listopada 2018

Wydział Nauk Technicznych i Ekonomicznych Państwowa Wyższa Szkoła Zawodowa im. Witelona w Legnicy

# Plan prezentacji

- 1. Aplikacje frontendowe
- 2. Reaktywne aplikacje frontendowe
- 3. Interludium: JavaScript
- 4. Przegląd rozwiązań
- 5. Podsumowanie

Aplikacje frontendowe

### Frontend?

Frontendem będziemy nazywali graficzny interfejs użytkownika (GUI) aplikacji webowej. W dużym uproszczeniu możemy wydzielić między innymi warstwę **prezentacji** oraz warstwę **interakcji**.

Pierwsza powinna przedstawiać dane przesłane z backendu aplikacji, druga - umożliwić zmianę tych danych.

### Frontend = HTML?

Najprostszy frontend? Będzie to zwykły HTML za pomocą którego można stworzyć praktycznie kompletny i funkcjonalny interfejs dla niektórych rozwiązań.

```
<! DOCTYPE html>
<html lang="en">
 <head>
   <meta charset="utf-8">
    <title>Homepage title</title>
  </head>
 <body>
    <h1>Title</h1>
   Content 
  </body>
</html>
```

### Frontend != HTML

Niestety szybko może się okazać, że sam HTML nie wystarczy i do pracy będzie trzeba zaprzegnąć *coś więcej*.

Po stronie serwera można wygenerować interfejs użytkownika za pomocą PHP (do czego zresztą początkowo przecież powstał) lub systemów szablonów dla większości języków programowania. Można wówczas skorzystać z dobrodziejstw instrukcji warunkowych, pętli i innych konstrukcji języka, które zautomatyzują budowanie frontendu (chociażby list i tabel).

### Zrzućmy to na klienta

Z czasem modne stało się przerzucanie odpowiedzialności za renderowanie frontendu z serwera aplikacji na użytkownika.

Ostateczną formą realizacji tej idei są SPA wykorzystające zewnętrzne API.

Wszystko oczywiście za pomocą JavaScriptu i jego dialektów.

### jQuery

```
$loading = $("#loading");
$form = $("#user-form',');
$button = $("#user-save-button");
$alertContent = $("#success-info > div");
$button.click(function() {
    $loading.show();
    url = "/api/user/" + $form.data("user-id");
    data = $form.serialize();
    $.post(url, data, function(response) {
        $loading.hide();
        $alertContent.html(response.message);
   });
});
```

### jQuery

Jeszcze kilka lat temu biblioteka jQuery była najprawdopodobniej najpopularniejszym frontendowym rozwiązaniem w internecie. Wciąż jest masowo stosowana, jednakże powoli wypierają ją reaktywne frameworki.

# \_\_\_\_\_

Reaktywne aplikacje frontendowe

# Przegląd rozwiązań

Obecnie warto się zainteresować jednym z poniższych:

- Angular
- React
- Vue.js

# Przegląd rozwiązań

Należy też na pewno wspomnieć o innych, może obecnie nieco mniej popularnych:

- Meteor
- Ember.js
- Backbone.js
- Aurelia
- Polymer



Jak należy rozumieć reaktywność w aplikacji webowej?

### Reaktywność?

Wyobraźmy sobie formularz z polem tekstowym, do którego użytkownik wpisuje wartość, którą chce znaleźć w tabeli poniżej.

### Reaktywność?

Korzystając z jQuery (lub nawet *vanilla JS*, trzeba obserwować element DOM i reagować na jego zmiany:

### Reaktywność?

Powyższe rozwiązanie działa, ale nie nazwałbym go dobrym. Ma też kilka mankamentów, między innymi:

- przy każdej zmianie w polu tekstowym pobiera nową listę wyników,
- za każdym razem czyści całą tabelę,
- miesza warstwy logiki biznesowej i prezentacji danych,
- zaczyna generować tzw. callback hell,
- i wiele innych.

### Reaktywność!

Poniżej mamy podobne formularz i tabelę, ale zbudowane przy pomocy Vue.js:

```
<div id="application">
   <form method="POST" class="search form">
      <input
         class="standard full width text input"
         placeholder="Enter searching phrase..."
         v-model="searchPhrase"
         autofocus
      >
   </form>
   {f result }}
     </div>
```

# Reaktywność!

```
export default {
    data() {
        return {
            fetchedResults: [],
            searchPhrase: "",
        }
    },
    computed: {
        results() {
            let phrase = this.searchPhrase
            return this.fetchedResults
                .filter(result => result.includes(phrase))
        }
    },
    mounted() {
        fetch("/api/results?" + this.searechPhrase)
            .then(response => {
                this.fetchedResults = response.data
            })
```

### Reaktywność!

### Co widać?

- zmienna searchPhrase jest obserwowana przez Vue i przy zmianie wartości pola tekstowego nastąpi automatyczne filtrowanie uprzednio pobranych danych,
- tabela jest renderowana na bieżąco w zależności od wyników,
- rozdziela warstwy logiki biznesowej i prezentacji danych,
- wykorzystujemy tutaj arrow functions, które między innymi zwiększają czytelność kodu.

# Programowanie reaktywne

Programowanie reaktywne to jeden z paradygmatyów programowania. Do tej pory na wykładach rozmawialiśmy między innymi o programowaniu strukturalnym, obiektowym, funkcyjnym czy imperatywnym.

# Programowanie reaktywne

Programowanie reaktywne opiera się na reakcjach na zdarzenia, które z grubsza można podzielić na trzy typy: interakcje od strony użytkownika, odpowiedzi serwera oraz zdarzenia wewnętrzne, systemowe.

Interludium: JavaScript

### Java w wersji Script

JavaScript jest obecnie jednym z najpopularniejszych, a jednocześnie jednym z najbardziej kochanych, znienawidzonych, nierozumianych, niedocenianych i przereklamowanych języków programowania.

Sam wstęp dużo mówi o tym jakiego rodzaju to język.

### npm

Podstawą środowiska deweloperskiego dla programisty JS będą między innymi:

- Node.js, czyli środowisko uruchomieniowe;
- npm, czyli manager pakietów;
- webpack, czyli bundler modułów
- Babel, czyli kompilator/transpilator

... i masa innych programów, które co kilka miesięcy się zmieniają.

### Warto wiedzieć #1: hot-reload

Korzystając z webpacka (ale także Grunta lub Gulpa) można skonfigurować projekt pod tzw. hot-reloading. Wówczas każda zmiana kodu w naszym IDE zostanie odnotowana przez system, przetworzona oraz na żywo wyświetlona w oknie przeglądarki.

Jedna z najwygodniejszych frontendowych funkcjonalności, którą warto zbadać.

### Warto wiedzieć #2: ES6

ES6, czyli ECMAScript 2016, to standard, który przyniósł wiele dobrego JavaScriptowi. Między innymi wprowadził funkcje strzałkowe, które upraszczają zapis, ale także zmienają zakres zmiennej this:

```
var self = this;
[4, 8, 15, 16, 23, 42].map(function(n) {
    return n * self.getMultiplier();
});
```

Na pewno wersja strzałkowa wygląda schludniej:

```
[4, 8, 15, 16, 23, 42].map(n => n * this.getMultiplier())
```

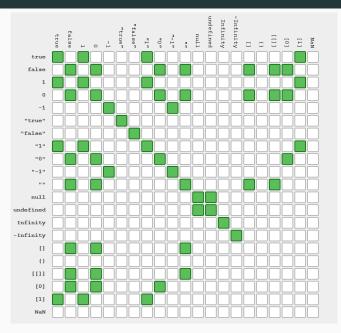
# Warto wiedzieć #3: TypeScript

Jedną z najczęściej wytykanych wad JavaScriptu jest jego rzekoma nieprzewidywalność w ujęciu typowania. Dla takich ludzi stworzono TypeScript, który umożliwia statyczne typowanie:

```
function sum(a, b) {
    return a + b
}

function sum(a: number, b: number): number {
    return a + b
}
```

# Warto wiedzieć #3: TypeScript



Przegląd rozwiązań

### **Angular**

Angular, poprzednio AngularJS, to opracowany przez Google frontendowy framework i obecnie jedno z dwóch najpopularniejszych rozwiązań tego typu na świecie.

### **Angular**

- wymaga TypeScriptu;
- model MVVM gwarantujący separację warstw;
- warstwa prezentacji danych oparta o ulepszony HTML;
- dwustronne wiązanie danych, a więcej reaktywne podejście;

### **Angular**

- posiada rozbudowany system wstrzykiwania zależności;
- gigantyczna społeczność i szczegółowa dokumentacja;
- każdy komponent definiuje się jako nowy folder z przynajmniej trzema plikami: \*.html, \*.ts i \*.scss;
- nowe wersje wychodza stosunkowo bardzo często i bywają ze sobą niekompatybilne.

### React

React to opracowany - tym razem - przez Facebooka frontendowy framework i obecnie również jedno z dwóch najpopularniejszych rozwiązań tego typu na świecie.

### React

- reklamuje się jako biblioteka, nie framework;
- wykorzystuje Virtual DOM;
- bardzo lekki i szybki;
- wymaga bardziej zaawansowanej wiedzy na temat JavaScriptu;
- jednostronne wiązanie danych;
- korzystając z React Native można budować aplikacje mobilne;

### React

wykorzystuje JSX:

```
const navigation = document.getElementById("navigation");
const user = {
    id: "cb0824c1-b88c-4965-b490-b6a9066aff44",
    name: "krewak"
};
const component = (
    <div>
        <i class="user icon"></i>
        { user.name }
    </div>
);
ReactDOM.render(welcomer, navigation);
```



Vue.js to opracowany głównie przez Evana You frontendowy framework i obecnie jedno z najbardziej rozwijających się rozwiązań tego typu.

# Vue.js

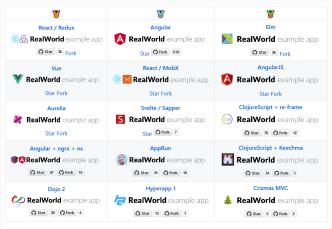
- szczegółowa dokumentacja;
- niewiele waży;
- może być stosowane jako import z CDN-a dla jednego komponentu lub pełna SPA;
- korzysta z Virtual DOM;

### Vue.js

- jedno- i dwustronne wiązanie danych;
- udostępnia możliwość tworzenia tzw. Vue components
- umożliwia zmianę języków i dialektów dla szablonów, stylów i skryptów;
- uważany za najłatwiejszy do nauczenia się.

**Podsumowanie** 

### W prawdziwym świecie



Work in Progress: Implement GraphQL + Apollo/Relay for the Node + React codebases | 📗 & Ember | 📗 & ClojureScript | 📗 & Mihrir II | 📗 & Vanilla JS (Web Components) | Angular 4 + / MobX | Angular JS 1.2 | Vanilla Backbonejs | Ractive | Praect | Quasar framework | Hydrating Vanilla JS (vaguely based on web components) | Blazor | Bridge.Spaf | Smol | Surplus wip | Slimjs | Nuxt.js | Typescript + Web Components

# Bibliografia i ciekawe źródła



https://realworld.io/ - naprawdę warto!



https://dzone.com/articles/react-vs-angular-vs-vuejs-a-complete-comparison-gu



Kod prezentacji dostępny jest w repozytorium git pod adresem https://bitbucket.org/krewak/pwsz-ppsi2



Wszystkie informacje dot. kursu dostępne są pod adresem http://pwsz.rewak.pl/kursy/6

