

NOWOCZESNY STOS JS W 60 MINUT

Wojciech Frącz, 2017

ep:zody

NOWOCZESNY STOS JS W 600 MINUT

Wojciech Frącz, 2017

ep:zody

O MNIE

Ph.D.



in progress



fracz
9,231
10 ● 56 ● 98



fracz@agh.edu.pl

CO TO BĘDZIE?

1. SPA

2. NARZĘDZIA

3. KOMPONENTY

**CO TO JEST SINGLE
PAGE APPLICATION
(SPA)?**

SINGE PAGE APPLICATION (KONTRPRZYKŁAD)

Serwer

Przeglądarka

 BAZA DANYCH



APLIKACJA (BACKEND)



 STRONA HTML Z
DANYMI

HTTP(S)



DODAJĘ BAJERY



AJAX

ASYNCHRONOUS JAVASCRIPT AND ~~XML~~ JSON

SINGLE PAGE APPLICATION

Serwer

Przeglądarka

 BAZA DANYCH



APLIKACJA (BACKEND)



</> DANE JSON

HTTP(S)
AJAX



 STRONA HTML



APLIKACJA (SPA)

SINGLE PAGE APPLICATION

Serwer

Przeglądarka

 BAZA DANYCH



APLIKACJA (MVC)



</> DANE JSON

HTTP(S)
AJAX
REST

 STRONA HTML



APLIKACJA (SPA)



CO TO BĘDZIE?

1. SPA

2. NARZĘDZIA

3. KOMPONENTY

NARZĘDZIA



NODE JS



- JS "na serwerze"
- JS uruchamiany "z konsoli"
- Budowanie aplikacji do przeglądarki

<https://nodejs.org>

NODE PACKAGE MANAGER (NPM)



- Zarządzanie zależnościami
- Uruchamianie przygotowanych skryptów

<https://www.npmjs.com/>

INICJALIZACJA APLIKACJI

```
$ npm init -y
```

```
Wrote to D:\projects\js-stack-60\pack
```

```
{  
  "name": "js-stack-60",  
  "version": "1.0.0",  
  "description": "",  
  "keywords": [],  
  "author": "",  
  "license": "ISC"  
}
```

ECMAScript 2015 (ES6)



- Klasy i dziedziczenie
- Arrow functions $x \Rightarrow x*2$
- Zmienne zakresowe `let`
- Promisy
- Moduły
- ...

<https://nodejs.org>



<http://steamingpenguin.deviantart.com/art/Internet-Explorer-Icon-256145520>



<https://slo-tech.com/forum/t217871/3149>

**JAK PISAĆ W ES6 JUŻ
DZIŚ?**

BABEL



- Transpilacja kodu
- Tłumaczy ES6 na ES5 (JS)
- Dzięki temu można pisać w ES6 już dziś

<https://babeljs.io/>

ES6 - PRZYKŁAD TRANSPILACJI

ES6

```
for (let x of [1, 2, 3]) {  
  console.log(`${x}^2 is ${x**2}`.  
}
```

1^2 is 1.
2^2 is 4.
3^2 is 9.

ES5

```
"use strict";  
  
var _arr = [1, 2, 3];  
for (var _i = 0;  
      _i < _arr.length; _i++) {  
  var x = _arr[_i];  
  console.log(x + "^2 is "  
    + Math.pow(x, 2) + ".");  
}
```

DODAJEMY BABEL DO PROJEKTU

```
$ npm install --save-dev babel-core babel-  
preset-es2015  
js-stack-60@1.0.0 D:\projects\js-stack-60  
+-- babel-core@6.24.1  
| +-- babel-code-frame@6.22.0  
| | +-- chalk@1.1.3  
| | | +-- ansi-styles@2.2.1  
| | | +-- escape-string-regexp@1.0.5  
| | | +-- has-ansi@2.0.0  
| | | | `-- ansi-regex@2.1.1  
...
```

DODAJEMY BABEL DO PROJEKTU

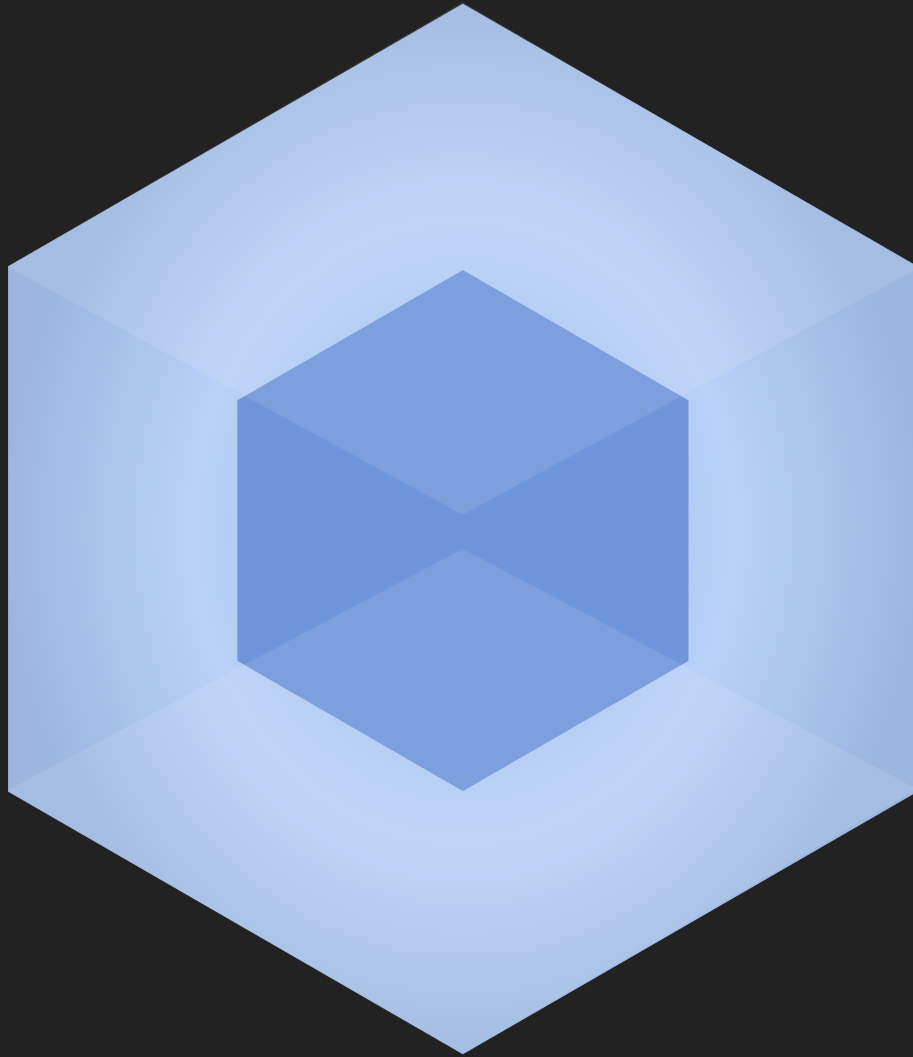
```
$ ls  
node_modules/ package.json
```

DODAJEMY BABEL DO PROJEKTU

```
$ ls node_modules
ansi-regex/
ansi-styles/
babel-code-frame/
babel-core/
babel-generator/
babel-helper-bindify-decorators/
babel-helper-call-delegate/
babel-helper-define-map/
babel-helper-explode-assignable-expression/
...
```

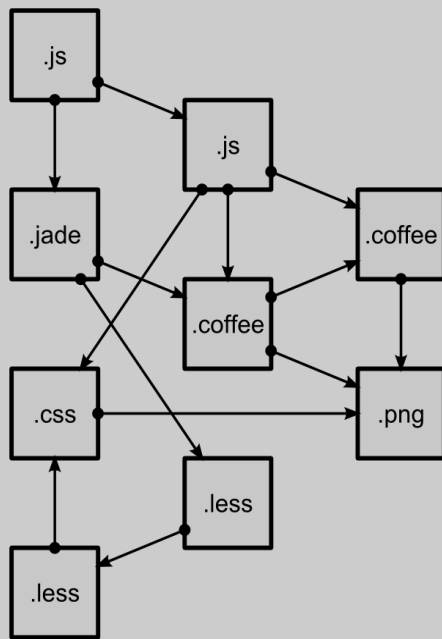
**JAK URUCHOMIĆ KOD
W PRZEGLĄDARCE?**

WEBPACK

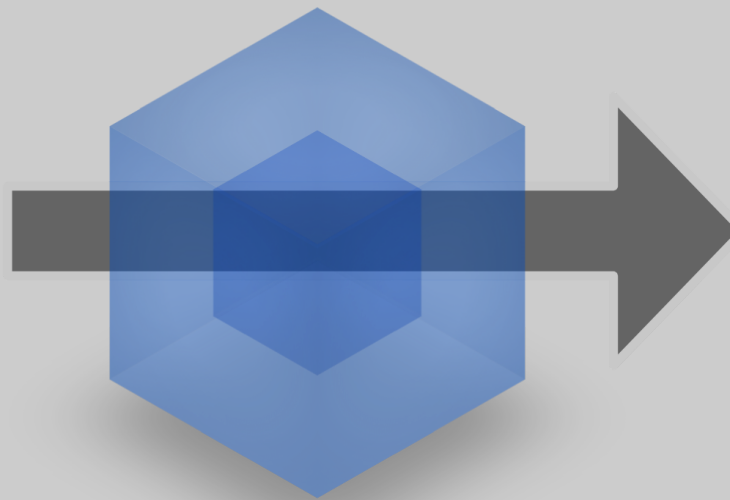


- Module bundler
- Dostarcza do przeglądarki kod w ujednoliconej, zrozumiałej formie

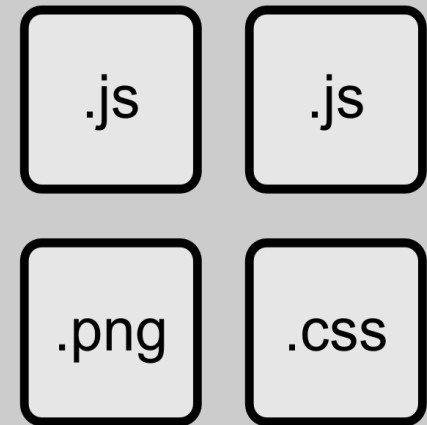
WEBPACK



modules
with dependencies



webpack
MODULE BUNDLER



static
assets

INSTALACJA WEBPACK

```
$ npm install --save-dev webpack babel-  
loader  
js-stack-60@1.0.0 D:\projects\js-stack-60  
+-- babel-loader@7.0.0  
| +-- find-cache-dir@0.1.1  
| | +-- commondir@1.0.1  
| | `-- pkg-dir@1.0.0  
| |   `-- find-up@1.1.2  
| |     +-- path-exists@2.1.0  
| |     `-- pinkie-promise@2.0.1  
| |       `-- pinkie@2.0.4  
...
```

KONFIGURACJA WEBPACK

\$

vi webpack.config.js

```
module.exports = {  
  entry: "./src/main.js",  
  output: {  
    path: __dirname + "/dist",  
    filename: "app.js",  
    publicPath: "/dist"  
  },  
  module: {  
    loaders: [  
      {  
        test: /\.js$/,  
        loader: "babel-loader",  
        exclude: /node_modules/  
      }  
    ]  
  }  
}
```

JAK BUDOWAĆ APLIKACJĘ?

SKRYPTY NPM

- Automatyzują proces budowania
- Definicja w `package.json`
- Dowolna komenda
- Widzą zależności zainstalowane przez NPM

```
"scripts": {  
  "build": "rm -fr dist && webpack"  
}
```

PACKAGE.JSON

```
{
  "name": "js-stack-60",
  "version": "1.0.0",
  "description": "",
  "keywords": [],
  "author": "",
  "license": "ISC",
  "scripts": {
    "build": "rm -fr dist && webpack"
  },
  "devDependencies": {
    "babel-core": "6.24.1",
    "babel-loader": "7.0.0",
    "babel-preset-es2015": "6.24.1",
    "webpack": "2.5.1"
  }
}
```

INDEX.HTML

\$

vi index.html

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <title>JS Stack 60</title>
</head>
<body>
<script src="dist/app.js"></script>
</body>
</html>
```


KOD W ES2015 (ES6)

```
$ vi src/main.js  
class Cat {  
    constructor() {  
        alert("Meow!");  
    }  
}  
  
new Cat();
```

STRUKTURA KATALOGÓW

```
js-stack-60/  
  node_modules/  
    ...  
    babel-core/  
    webpack/  
    ...  
  src/  
    main.js  
  index.html  
  package.json  
  webpack.config.js
```

<https://github.com/fracz/js-stack-60/tree/step-1>

BUDUJEMY APLIKACJĘ

```
$ npm run build
```





CO TO BĘDZIE?

1. SPA

2. NARZĘDZIA

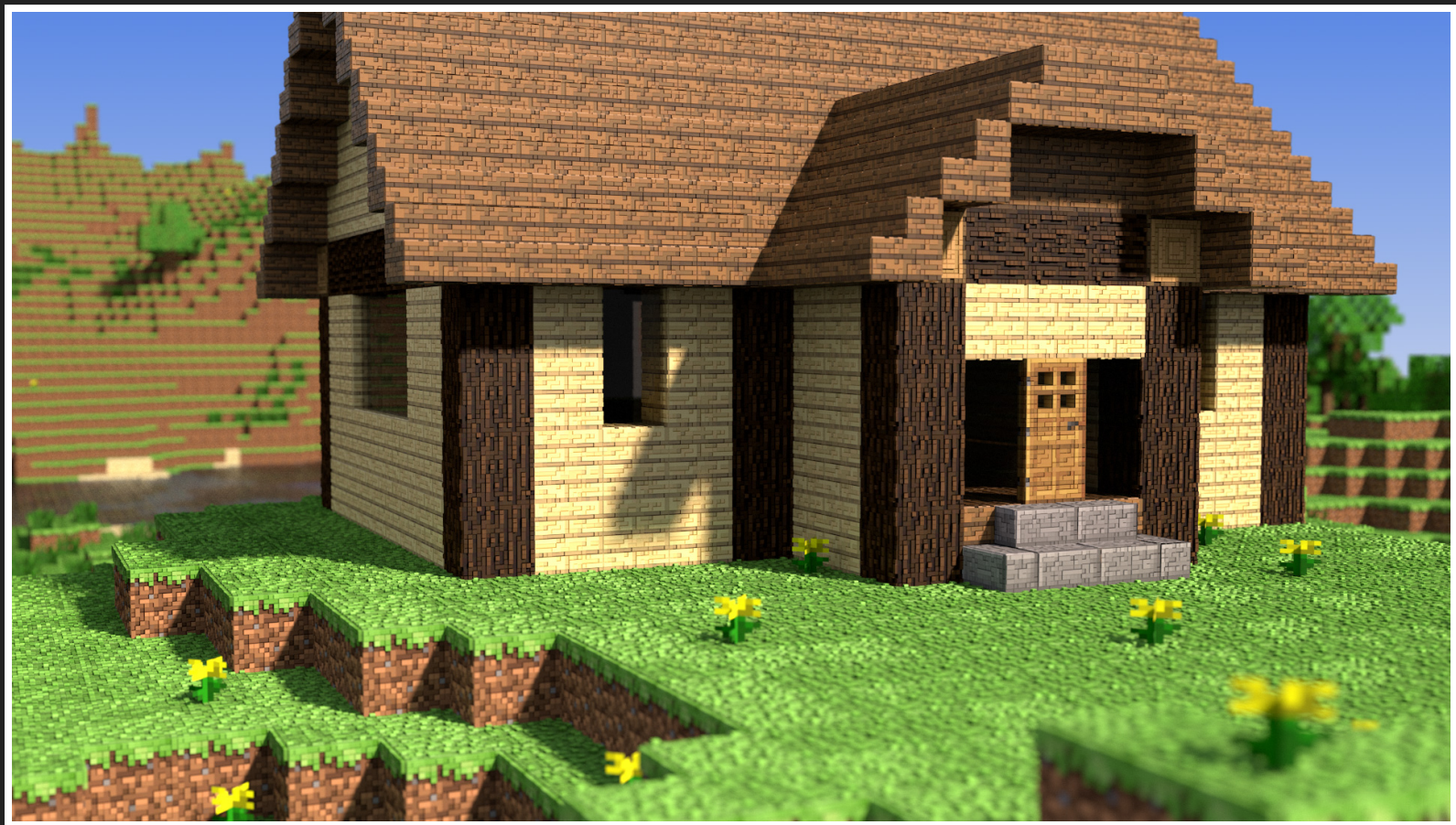
3. KOMPONENTY

CO TO JEST KOMPONENT?



- Struktura (HTML)
- Zachowanie i stan (JS)
- Wygląd (CSS)
- Jedna funkcjonalność (SRP)

APLIKACJA ZŁOŻONA Z KOMPONENTÓW



**JAK
ZAIMPLEMENTOWAĆ
KOMPONENTY?**

VUE.JS



- The Progressive JavaScript Framework
- SPA
- Reaktywny

<https://vuejs.org/>

DODAWANIE VUE.JS DO PROJEKTU

```
$ npm install --save vue  
js-stack-60@1.0.0 D:\projects\js-stack-60  
`-- vue@2.3.2
```

DODAWANIE VUE.JS DO PROJEKTU

```
$ npm install --save-dev vue-loader vue-  
template-compiler css-loader style-loader  
js-stack-60@1.0.0 D:\projects\js-stack-60  
+-- css-loader@0.28.0  
| +-- css-selector-tokenizer@0.7.0  
| | +-- cssesc@0.1.0  
| | +-- fastparse@1.1.1  
| | `-- regexpu-core@1.0.0  
| +-- cssnano@3.10.0  
| | +-- autoprefixer@6.7.7  
...  

```

NAPISZMY PIERWSZY KOMPONENT

- Napisaliśmy pierwszy komponent
- Zainicjalizowaliśmy framework Vue
- Dodaliśmy do `index.html` miejsce, gdzie będzie działać aplikacja
- Nauczyliśmy webpack jak ładować komponenty Vue

<https://github.com/fracz/js-stack-60/tree/step-2>

<http://gph.is/1UB39zl>

**CZY MUSZĘ TO TAK W
KÓŁKO BUDOWAĆ?**

I ODŚWIEŻAĆ...

I ZNOWU BUDOWAĆ...

HOT RELOAD



- Odświeżanie zmian w przeglądarce na podstawie zmian w kodzie
- Automatyczne!

<https://www.genuitec.com/products/webclipse/features/hotreload-code/>

INSTALACJA WEBPACK-DEV-SERVER

```
$ npm install --save-dev webpack-dev-server
js-stack-60@1.0.0 D:\projects\jsstack
`-- webpack-dev-server@2.4.5
   |-- ansi-html@0.0.7
   |-- compression@1.6.2
   |  |-- accepts@1.3.3
   |  |  `-- negotiator@0.6.1
   |  |-- bytes@2.3.0
   |  |-- compressible@2.0.10
   |  |  `-- mime-db@1.27.0
   ...
```

NPM RUN DEV

```
...  
"scripts": {  
  "build": "rm -fr dist && webpack",  
  "dev": "webpack-dev-server --hot --inline --open"  
},  
...
```

<https://github.com/fracz/js-stack-60/tree/step-3>

CZYM JEST REAKTYWNOŚĆ?

```
<template>
  <div>
    <h1>{{ title }}</h1>
    <input type="text" v-model="title">
  </div>
</template>

<script>
  export default {
    data() {
      return {
        title: 'Epizody'
      }
    }
  };
</script>
```


NASZA APLIKACJA (HTML)

```
<form @submit.prevent="saveMe()">
  <fieldset>
    <legend>Zarejestruj się tutaj</legend>
    <div>
      <label>Imię i nazwisko</label>
      <input type="text" v-model="newParticipant">
    </div>
    <div>
      <button>Zapisz mnie</button>
    </div>
  </fieldset>
</form>
```

NASZA APLIKACJA (JS)

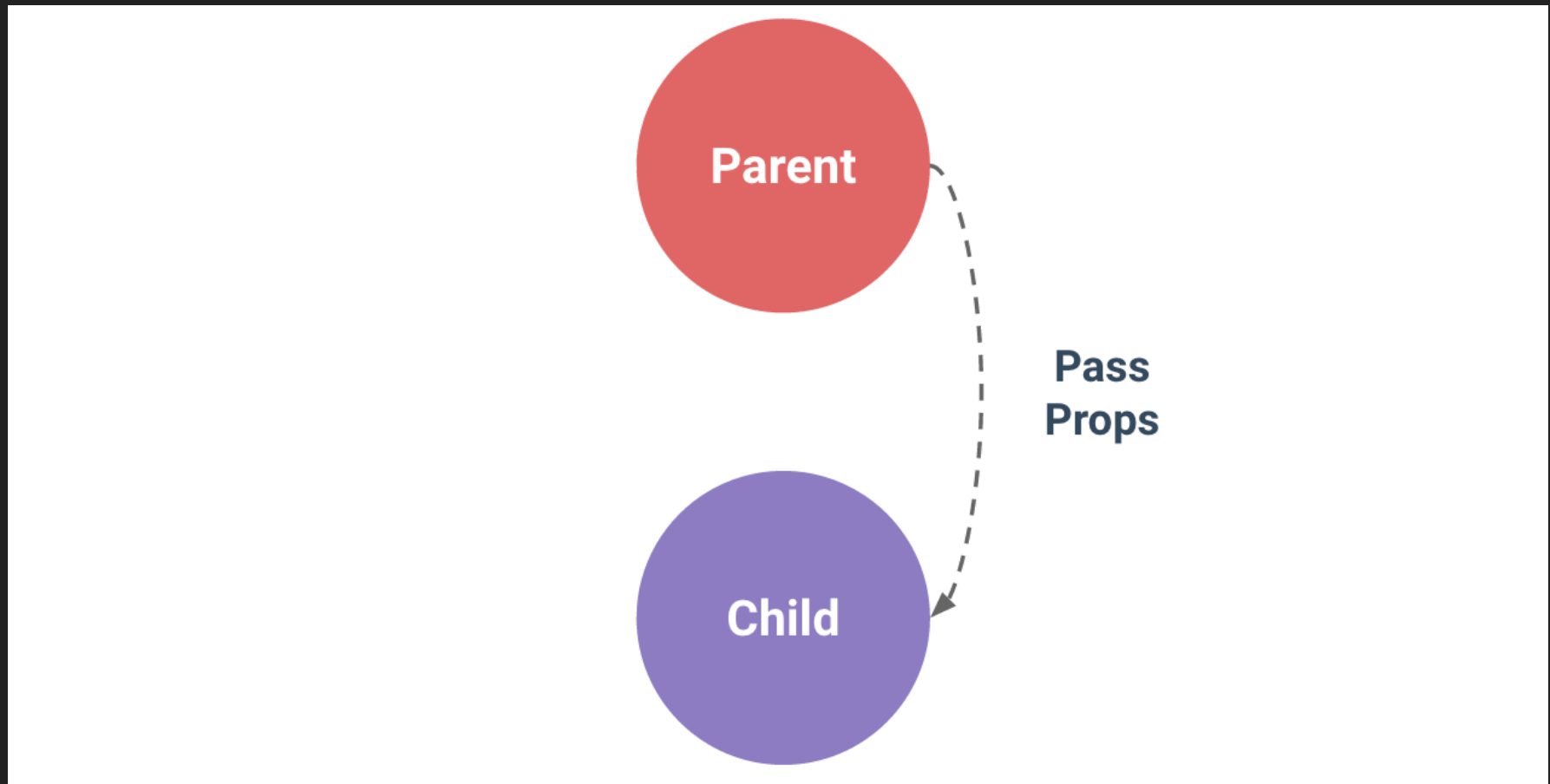
```
data() {  
  return {  
    title: 'Epizody',  
    newParticipant: '',  
    savedParticipants: []  
  }  
},  
methods: {  
  saveMe() {  
    if (this.newParticipant) {  
      this.savedParticipants.push(this.newParticipant);  
      this.newParticipant = '';  
    } else {  
      alert('You need to give us a name!');  
    }  
  }  
}  
}
```

A MIAŁO BYĆ SRP!

<https://github.com/fracz/js-stack-60/tree/step-4>

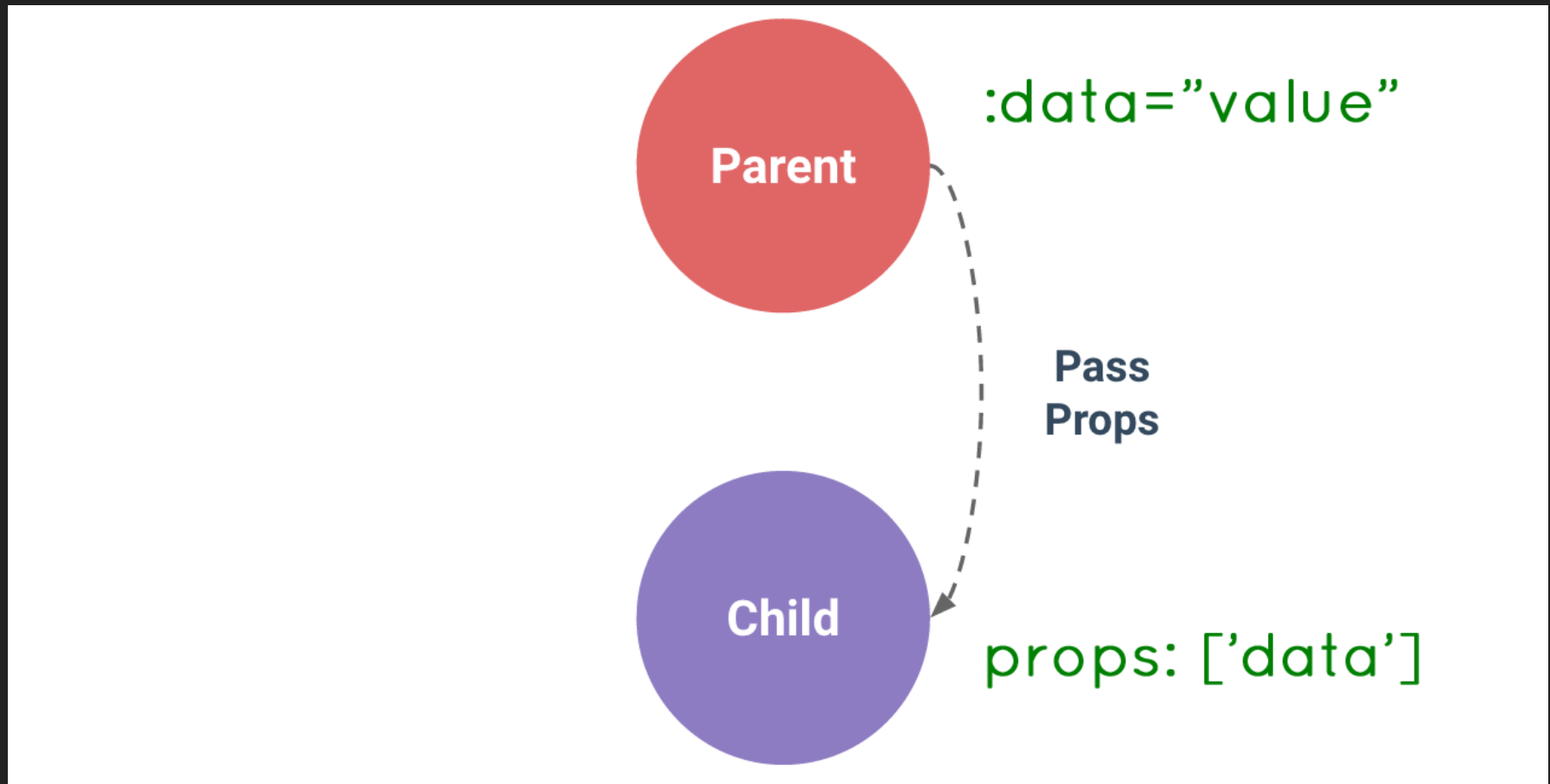
<https://www.slideshare.net/techexeter/enjoy-the-vuejs>

HIERARCHIA KOMPONENTÓW



<https://vuejs.org/v2/guide/components.html>

HIERARCHIA KOMPONENTÓW



CZY MAMY JESZCZE CZAS?

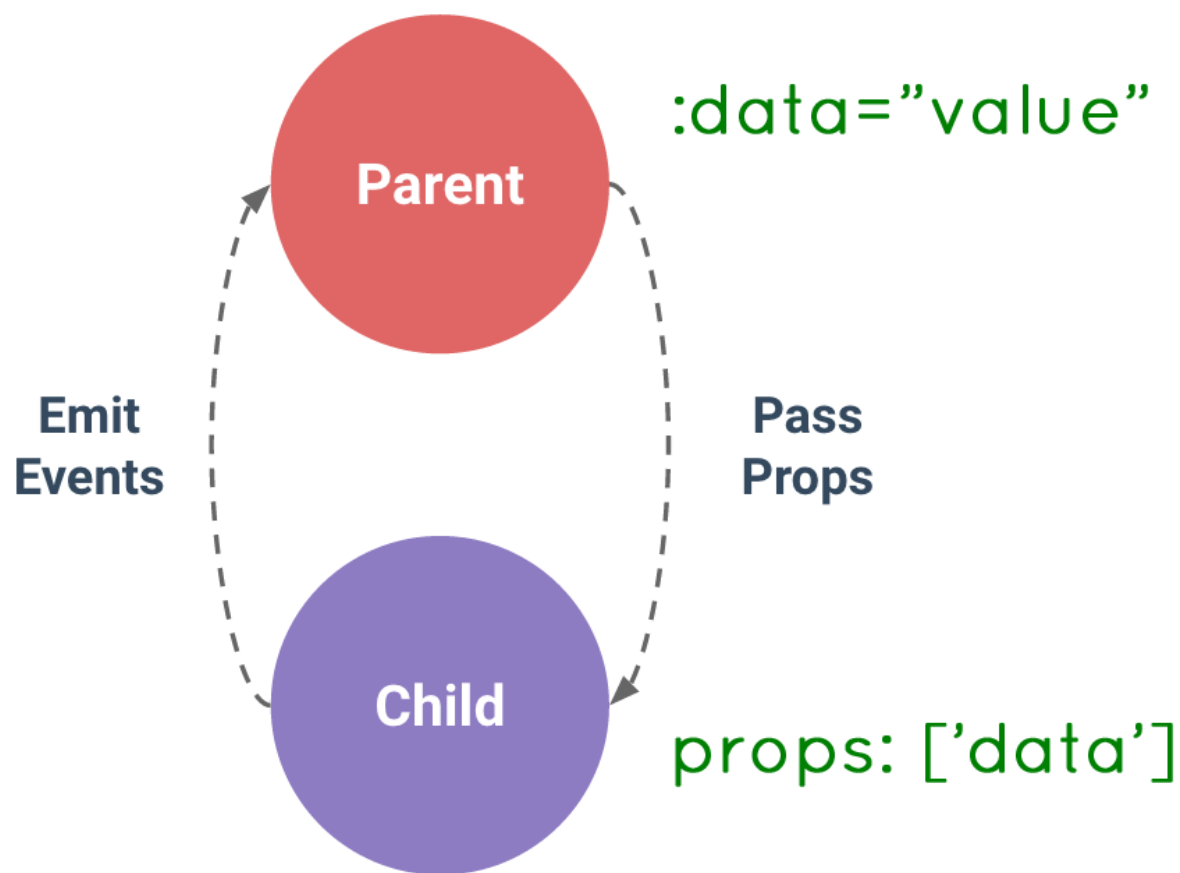


ZDEKOMPONUJMY APLIKACJĘ

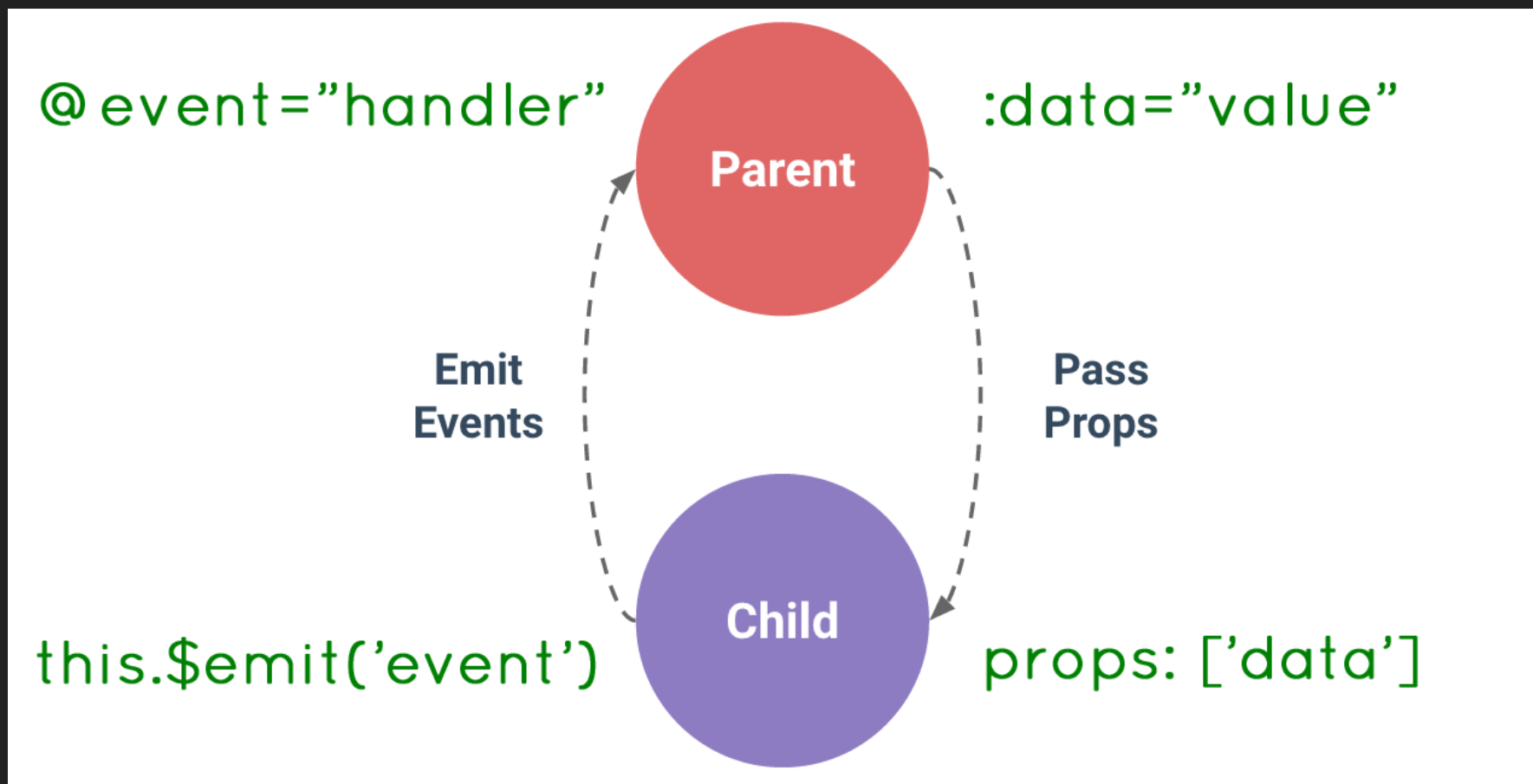
- Przenieśliśmy listę osób do komponentu
- Komponent nadrzędny przekazał do niego listę osób do wyświetlenia
- Dodaliśmy do niego więcej funkcjonalności
- Napisaliśmy styl scoped działający tylko w obrębie danego komponentu
- Użyliśmy komponentu w aplikacji

<https://github.com/fracz/js-stack-60/tree/step-5>

APLIKACJA ZŁOŻONA Z KOMPONENTÓW



APLIKACJA ZŁOŻONA Z KOMPONENTÓW



CZY MAMY JESZCZE CZAS?



WPROWADŹMY JESZCZE JEDEN KOMPONENT

- Przenieśliśmy formularz dodawania osób komponentu
- Komponent nadrzędny otrzymuje od niego nową osobę w postaci zdarzenia
- Walidacja została zamknięta w komponencie tak, że parent nie musi o nią dbać
- Użyliśmy komponentu w aplikacji

<https://github.com/fracz/js-stack-60/tree/step-6>

CZY MAMY JESZCZE CZAS?



USUWANIE ZAPISANYCH OSÓB

- Utworzyliśmy osobny komponent na element listy
- Jako wejście przyjmuje on zgłoszoną osobę
- Jako wyjście dostarcza zdarzenie kliknięcia w przycisk "Usuń"

<https://github.com/fracz/js-stack-60/tree/step-7>

<http://gph.is/XKrquS>

CZEGO NIE OMÓWILIŚMY?

- Budowanie wersji produkcyjnej
- Minifikacja kodu
- Generatory aplikacji
- Testowanie jednostkowe komponentów
- Lazy loading
- Tree shaking
- Routing (widoki)
- Animacje
- Frameworki CSS
- Preprocesory CSS
- Dyrektywy
- Filtry
- State management pattern (Vuex, Redux)

- Testy E2E
- CSS Autoprefixer (PostCSS)
- Integracja z niereaktywnymi bibliotekami (np. jQuery)
- Virtual DOM
- One-way vs Two-way binding
- ES Lint
- Yarn
- Promisy
- Mixiny
- Computed properties
- Watchers
- Modifiers
- HTML5 Push State
- Server-side rendering
- Asynchroniczne wywołania `await`
- Różnice w `this` w Arrow functions

- REST
- JWT
- ES7
- Typescript
- Internacjonalizacja
- Local storage

CZEGO NIE OMÓWILIŚMY?

- Budowanie wersji produkcyjnej
- Minifikacja kodu
- Generatory aplikacji
- Testowanie jednostkowe komponentów
- Lazy loading
- Tree shaking
- Routing (widoki)
- Animacje
- Frameworki CSS
- Preprocesory CSS
- Dyrektywy
- Filtry
- State management pattern (Vuex, Redux)
- Testy E2E
- CSS Autoprefixer (PostCSS)
- Integracja z niereaktywnymi bibliotekami (np. jQuery)
- Virtual DOM
- One-way vs Two-way binding
- ES Lint
- Yarn
- Promisy
- Mixiny
- Computed properties
- Watchers
- Modifiers
- HTML5 Push State
- Server-side rendering
- Asynchroniczne wywołania `await`
- Różnice w `this` w Arrow functions
- REST
- JWT
- ES7
- Typescript
- Internacjonalizacja
- Local storage

PYTANIA?



DZIĘKUJĘ!



Prezentacja

<https://fracz.github.io/js-stack-60>

Źródła

<https://github.com/fracz/js-stack-60>

Wojciech Frącz

fracz@agh.edu.pl