Typowanie w JS

Marcin Kwiatkowski

```
let a = 5
...
a = {
    "name": „Jan”,
    "age": 5
}
…
```

Czym jest a?

```
function Button(props) {
    return (
        <button onClick={props.onClick}>
            {props.text}
        </button>
    )
}
```

Typowanie w JS

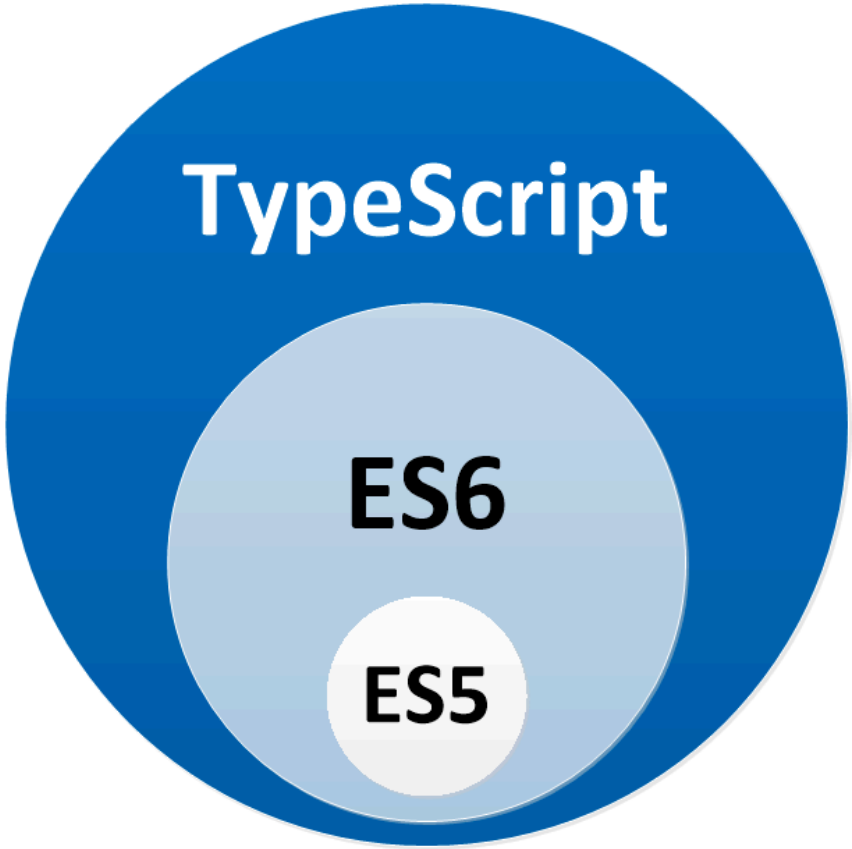
- JavaScript ma typowanie dynamiczne
  - Typ wartości określany jest w trakcie działania programu
- Brakuje typowania statycznego, które pozwala kontrolować integralność aplikacji
- Ale również…
  - Typowanie statyczne daje więcej możliwości dla IDE (podpowiedzi)
  - O potencjalnych błędach dowiadujemy się wcześniej niż w runtime'ie
  - W świecie Javy to nic nadzwyczajnego, tam wszystko jest typowane z automatu :)

Typowanie w JS

- **TypeScript**
  - Microsoft
  - Dobre wsparcie w VSC
  - https://typescriptlang.org
- **Flow**
  - Facebook
  - https://flow.org

TypeScript

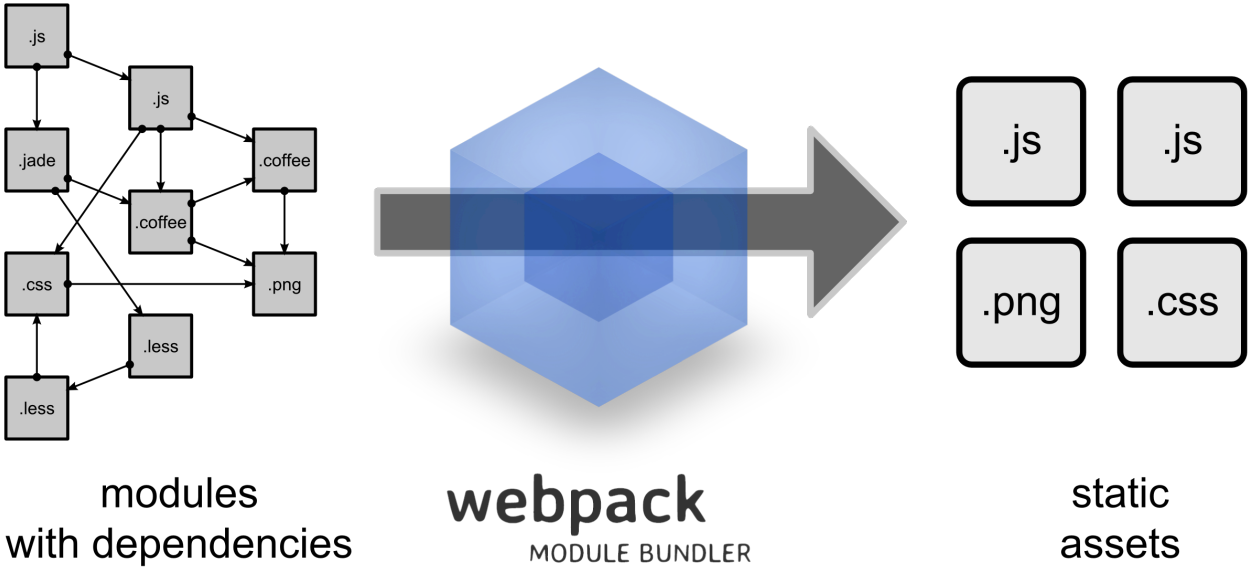
- Stworzony jako nadzbiór JS
- Bardzo popularny na rynku
- Z założenia wspiera migracje etapami (część w JS, część w TS)



TypeScript a CRA

Jak uruchamiany w przeglądarce jest kod pisany w ES6
w Create React App?

TypeScript a CRA



TypeScript a CRA

https://github.com/TypeStrong/ts-loader

Rozszerzenia plików

- *.ts
- *.tsx

Typing (*.d.ts)

- Z poziomu kodu w JS możemy korzystać z kodu w TS.
- Z poziomu kodu w TS możemy również korzystać z JS. Musimy jednak dostarczyć dodatkowe informacje na temat używanych typów.
- Pliki *.d.ts są albo dostarczane w paczce npm albo trzeba je pobrać osobno (@types/nazwa-paczki)

```
Could not find a declaration file for module 'react'.
'/Users/Marcin/iSA/Repozytoria/jfdz11-typowanie/contacts/node_modules/react/index.js'
implicitly has an 'any' type.
  Try `npm install @types/react` if it exists or add a new declaration (.d.ts) file
containing `declare module 'react';` ts(7016)
```

Peek Problem No quick fixes available

Podstawowe typy

- boolean
- number
- string
- Array
- enum
- any
- void
- null & undefined

string

```
const str = „tekst"
const str: string = "tekst"

let str = "drugi tekst"
str = "inny tekst"
```

boolean

Jak utworzyć zmienną typu *boolean*?

```
const a = true
const a: boolean = true

let a = true
let a: boolean = true
```

number

```
const b = 1
const b: number = 1

let b = 4.5
b = 6
b = ”4.3” // ERROR
```

enum

```
enum IssueStatus {
    NEW: 0,
    IN_PROGRESS: 1,
    CLOSED: 2
}

const status = IssueStatus.NEW
```

Array

```
const list: number[] = [1, 3, 4]
const list: Array<number> = [1, 3, 4]
```

any

- Określenie, że zmienna jest dowolnego typu
- Symulacja domyślnego działania JS
- Bardzo przydatne w procesie migracji
- W praktyce dążymy, aby eliminować any na ile jest to możliwe

```
const objects[]: any[] = ...
```

void

- Wskazuje, że funkcja nic nie zwraca
- W pewnym sensie przeciwieństwo any

```
componentDidMount(): void {
}
```

Interface

- Określa jak wygląda obiekt (recepta)

```
interface Issue {
    id: string
    title: string
}

const issue: Issue = {}   ???
const issue: Issue = null ???
```

```
const issue: Issue = { "id": "33", "title": "Nowy feature”, „closed”: true}
```

Interface

```
interface ButtonProps {
    text: string
    onClick: () => void
}

function Button(props: ButtonProps) {
    return (
        <button onClick={props.onClick}>
            {props.text}
        </button>
    )
}
```

Konfiguracja - tsconfig.json

- Konfiguracja TypeScripta znajduje się w pliku tsconfig.json (ten sam poziom co src).
- https://www.typescriptlang.org/docs/handbook/tsconfig-json.html

Zadanie

- Zmień nazwę pliku `ContactElement.jsx` na `ContactElement.tsx`
- Zdefiniuj interfejs do propsów komponentu `ContactElement`
- Wskaż, że zmienna props jest zgodna z utworzonym interfejsem

Zadanie

Zmień na TS-a komponent `ContactList`

Zadanie

Zmień na TS-a plik z funkcjami `contactActions.js`

Zadanie

- Pozbądźmy się `any` przy `contact` !

- Utwórz plik `model.ts` w katalogu `src`
- Zdefiniuj interfejs Contact. Co o nim wiemy?
- Zmień `any` na utworzony interfejs

Zadanie

- Pozbądźmy się *any* przy `contactsContext` !

- Zmień nazwę ContactsContext.js na ContactsContext.ts
- Popraw wszystkie błędy wskazywane przez VSC

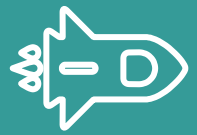
Zadanie

Zmień na TS-a komponent `ContactForm`

Zadanie

- Zmień nazwę pliku `ContactElement.jsx` na `ContactElement.tsx`
- Zdefiniuj interfejs do propsów komponentu `ContactElement`
- Wskaż, że zmienna props jest zgodna z utworzonym interfejsem

Ważne!



Przepisz na TypeScripta swój projekt realizowany w ramach infoShare Academy !!!

Znajomość TypeScripta będzie bardzo dobrym asem w rękawie!

Ciekawe linki

- https://www.typescriptlang.org/docs/handbook/basic-types.html
- http://www.typescriptlang.org/play/
- https://code.visualstudio.com/docs/typescript/typescript-tutorial
- https://typeofweb.com/2016/07/11/typescript-czesc-1/

Dzięki

LinkedIn: https://www.linkedin.com/in/mkwiatko

- Trener infoShare Academy
- Full Stack Developer w Bright Inventions