

Wrocław, dnia 09.06.2018 roku

Prezentacja końcowa

Studia podyplomowe: Programista front-end z Angular

Student: Piotr Ostrowski

Plan prezentacji

1. Wstęp
2. Cel i zakres wybranego
3. Opis rozwiązań
4. Prezentacja działania
5. Podsumowanie oraz wnioski z realizacji projektu
6. Plany rozwojowe na przyszłość

1. Wstęp

Na studiach podyplomowych Programista front-end z AngularJs jako zaliczeniowy projekt wybrałem:

Strona internetowa z użyciem routingu

Realizowany pod opieką Wojciecha Medyńskiego.

Celem jest poznanie i implementacja technik umożliwiających tworzenie aplikacji internetowych z użyciem frameworka Angular.

2. Cel i zakres projektu

W projekcie strony internetowej uwzględniłem następujące rozwiązania:

- Stylowanie Bootstrap
- Routing, obsługa błędu 404
- Implementacja mapy google
- Walidacja formularza z przestaniem danych do okna modalnego
- Galeria zdjęć z obsługą sterowania
- Paginacja

3. Opis rozwiązań

Stylowanie Bootstrap

Framework Bootstrap zawiera zestaw przydatnych narzędzi ułatwiających tworzenie interfejsu graficznego stron oraz aplikacji internetowych i stosowany jest do stylizacji elementów jak teksty, formularze, przyciski, wykresy, nawigacje i innych komponentów wyświetlanych na stronie

Włączenie stylowania polegało na instalacji paczki w npm i aktualizacji pliku `angular.json` w sekcji `styles`.

3. Opis rozwiązań

Stylowanie Bootstrap

Navbar

[Home](#) [Do pobrania](#) [Profil](#) [Media](#) [Kontakt](#)



Card title

This is a longer card with supporting text below as a natural lead-in to additional content. This content is a little bit longer.

Last updated 3 mins ago



Card title

This card has supporting text below as a natural lead-in to additional content.

Last updated 3 mins ago



Card title

This is a wider card with supporting text below as a natural lead-in to additional content. This card has even longer content than the first to show that equal height action.

Last updated 3 mins ago

FOOTER CONTENT

LINKS

LINKS

LINKS

3. Opis rozwiązań

Routing, obsługa błędu 404

W module aplikacji zadeklarowano stałą `appRoutes`, typu `Routes`, która przechowuje tablicę ścieżki i odpowiadającemu jej modułowi.

```
const routes: Routes = [ { path: 'sciezka', component: DestComponent } ];
```

Realizację błędu 404 (nieodnaleziona strona) zrealizowana jest przez parametr `***`

Reprezentacja routingu w szablonie odbywa się poprzez

```
<header></header>
```

```
    <router-outlet></router-outlet>
```

```
<footer></footer>
```

3. Opis rozwiązań, Routing, obsługa błędu 404

Navbar

[Home](#) [Do pobrania](#) [Profil](#) [Media](#) [Kontakt](#)



Strona nie istnieje

3. Opis rozwiązań

Implementacja mapy Google – Angular Google Maps

Zastosowanie mapy w projekcie wymaga uzyskania klucza API Google Maps.

Poprzez import paczki AgmCoreModule uzyskujemy możliwość osadzenia mapy w szablonie poprzez znacznik `<agm-map>` z możliwością konfiguracji punktów poprzez `<agm-marker>`.

```
<agm-map [latitude]="lat" [longitude]="lng" [zoom]="zoom"> //konfiguracja widoku i powiększenia
```

```
  <agm-marker *ngFor = "let mrk of markers; let i = index
```

```
    [latitude]="mrk.lat"
```

```
    [longitude]="mrk.lng">
```

```
</agm-marker>
```


```
</agm-map>
```

Konfiguracja współrzędnych markera następuje po przypisaniu do zmiennej `markers` tablicy współrzędnych długości i szerokości geograficznej punktów.


```
markers: marker[] = [ { lat: 51.113033, lng: 16.981337 }, {lat:, lng: }, {...} ]
```

3. Opis rozwiązań - Angular Google Maps


[Home](#) [Do pobrania](#) [Profil](#) [Media](#) [Kontakt](#)

 **Write to us:**

We'll write rarely, but only the best content.




Imie



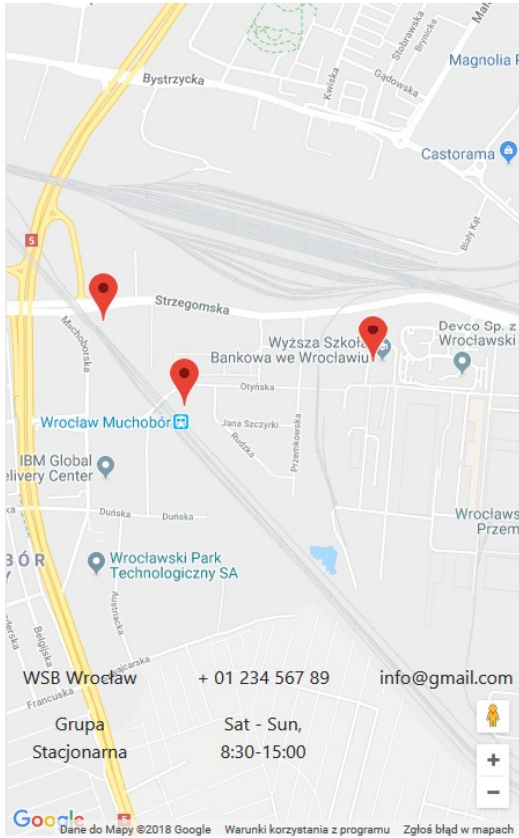
Email

Temat



Loem ipsum dolor sit amet consectetur
adipiscing elit.
Error, quis praesentium quas nisi nulla itaque
dolorum, enim dolorem reiciendis accusamus
corporis
aliquid vero dolores impedit repellat nostrum

Treść



Map showing Wrocław, Poland, with several red location pins. Labels on the map include: Bystrzycka, Strzegomska, Bankowa we Wrocławiu, Wyższa Szkoła, Devco Sp. z Wrocławski, Wrocław Muchobór, IBM Global Delivery Center, Wrocławski Park Technologiczny SA, WSB Wrocław, Grupa Stacjonarna, + 01 234 567 89, info@gmail.com, Sat - Sun, 8:30-15:00.

3. Opis rozwiązań

Walidacja formularza reaktywnego z przestaniem danych do okna modalnego

Formularze reaktywne pozwalają na kontrolę i zarządzanie bezpośrednio w klasie komponentu.

Tworzone jest drzewo obiektów i odpowiadające mu elementy formularza.

Do zarządzania formularzem wymagane są klasy FormControl i FormGroup z paczki @angular/forms, które pozwalają na śledzenie zmian oraz walidację.

Odzwierciedleniem pól formularza, jest deklaracja obiektu modelForm typu FormGroup.

Do obsługi błędów, wykorzystujemy obiekt formErrors zawierający pola formularza i odpowiadający mu validationMessages z opisem błędów dla pola input wymagającego walidacji.

```
formErrors = { imie: ' ', ... }  
validationMessages = { imie: {  
    required: 'Imie wymagane',  
    minlength: 'Minimum 3 znaki'}, ... }
```

W metodzie ngOnInit() deklarowany jest obiekt zawierający wyszczególnione pola z zakresem walidacji dostępnych z klasy Validators,

```
this.modelForm = this.formBuilder.group({  
    imie: ['Jan', [Validators.minLength(3), Validators.required]], ... })
```

następnie subskrybowana wartość pól przy każdej jej zmianie uruchamia metodę onChangeValue().

3. Opis rozwiązań - Walidacja formularza

```
this.modelForm.valueChanges.subscribe((value) => {  
  this.onChangeValue();  
});
```

Metoda `onChangeValue` zawiera przyporządkowanie pól formularza do właściwości klasy.
`this.imie = form.controls.imie.value;`

Każda zmiana pola powoduje zmianę wartości właściwości obiektu typu *FormControl* takie jak
 `$pristine` – TRUE jeśli nie nastąpiła interakcja,
 `$dirty` – TRUE jeśli już była interakcja,
 `$valid` – TRUE jeśli pole jest poprawne.

Obiekt ten posiada właściwość `$error` przyjmujący tokeny takie jak *maxlength*, *required*, *minlength*, *email* itd. które porównywane są z *validationMessages*

Wysłanie formularza możliwe jest gdy wartość `modelForm.valid` jest true (wszystkie pola mają właściwość `$valid` na TRUE).

Wysłanie formularza uruchamia metodę `openDialog(modelForm)` gdzie przekazywany jest formularz z wartościami wpisanymi pól.

```
let fileNameDialogRef = this.dialog.open(..., {  
  width: '550px',  
  data: { imie: this.imie, nazwisko: this.nazwisko, mail: this.mail } });
```

3. Opis rozwiązań - Walidacja formularza

Błędy walidacji

✉ Write to us:
We'll write rarely, but only the best content.

Imię **Imie wymagane**

Nazwisko **Minimum 3 znaki**

Email **niepoprawny adres email**

Temat

Treść

Wyślij

Okno modalne przy poprawnym formularzu

Cześć Jan

Gratulacje, poprawnie wypełniłeś formularz

Twoje dane

Jan
Kowalski
jan.kowalski@domena.pl

Ok

3. Opis rozwiązań

Galeria zdjęć z obsługą sterowania

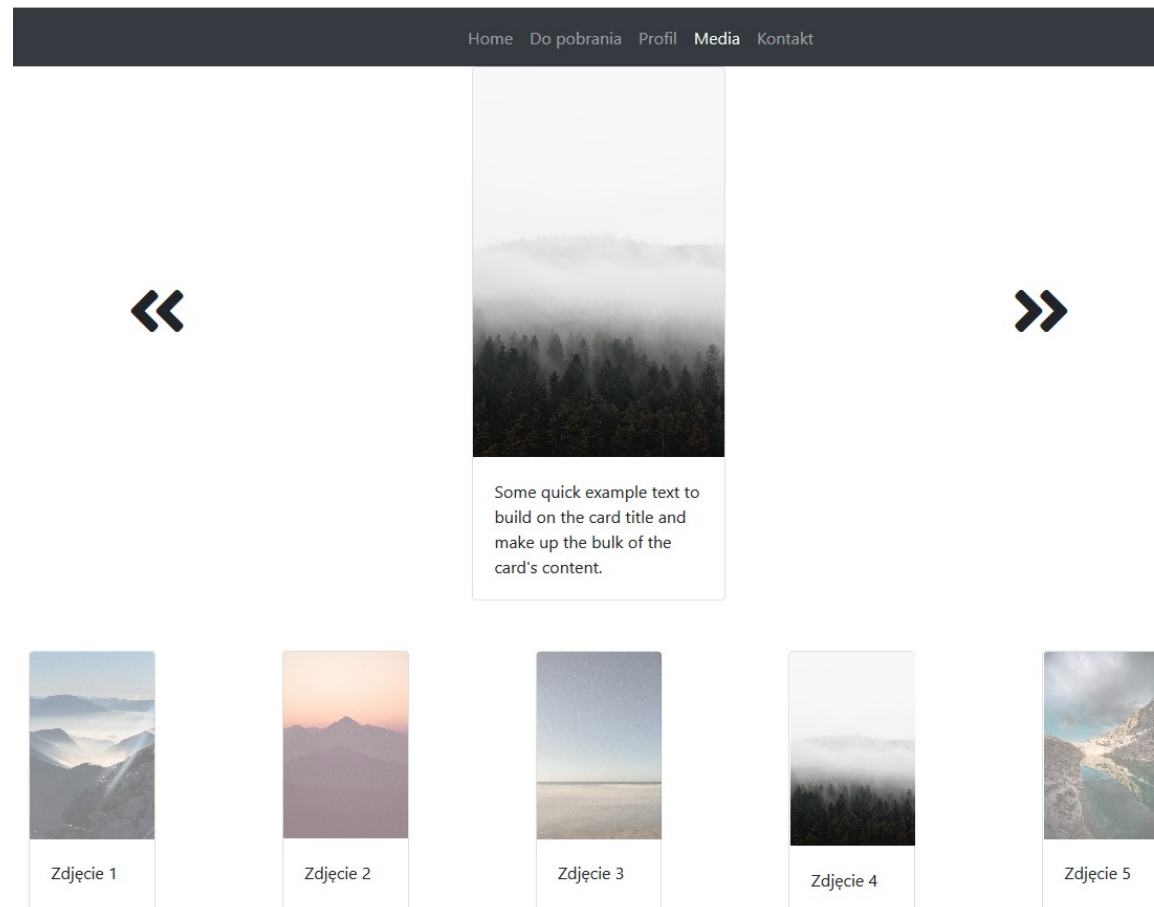
Wyświetlanie zdjęć realizowane jest poprzez odczyt wartości tablicy elementów zawierającą ścieżki do plików jpg i podstawianie jej w szablonie w elemencie [src].

Aktywne zdjęcie ma dodawaną klasę .active, która zadeklarowana jest w css jako opacity 1;

Wyświetlanie galerii zmienia się automatycznie poprzez wywołanie funkcji selectNext() z funkcji setInterval() z czasem 3 sekundy.

Galeria zdjęć wyświetlana jest za pomocą instrukcji ngFor w szablonie a na każdy element nałożone jest zdarzenie (click), które wywołuje funkcję wyświetlającą wybrane zdjęcie.

3. Opis rozwiązań - Galeria zdjęć



3. Opis rozwiązań

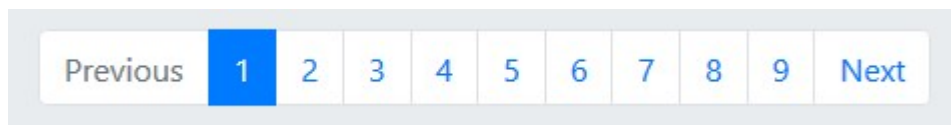
Paginacja

Na potrzeby projektu utworzona została funkcja generująca przypadkowe wyrazy z zakresu liter / cyfr podanych w tablicy, która losowo generuje wystąpienia w zakresie 5 do 105.

Wystąpienia słów są numerowane, aby umożliwić przetestowanie rozwiązania.

Wybranie następnego numeru paginacji wywołuje utworzenie nowej tablicy z wybraniem kolejnej dziesiątki elementów metodą tablicową slice, gdzie początkiem jest cyfra paginacji pomnożona razy 10 i wybrane następne 10 elementów jako koniec

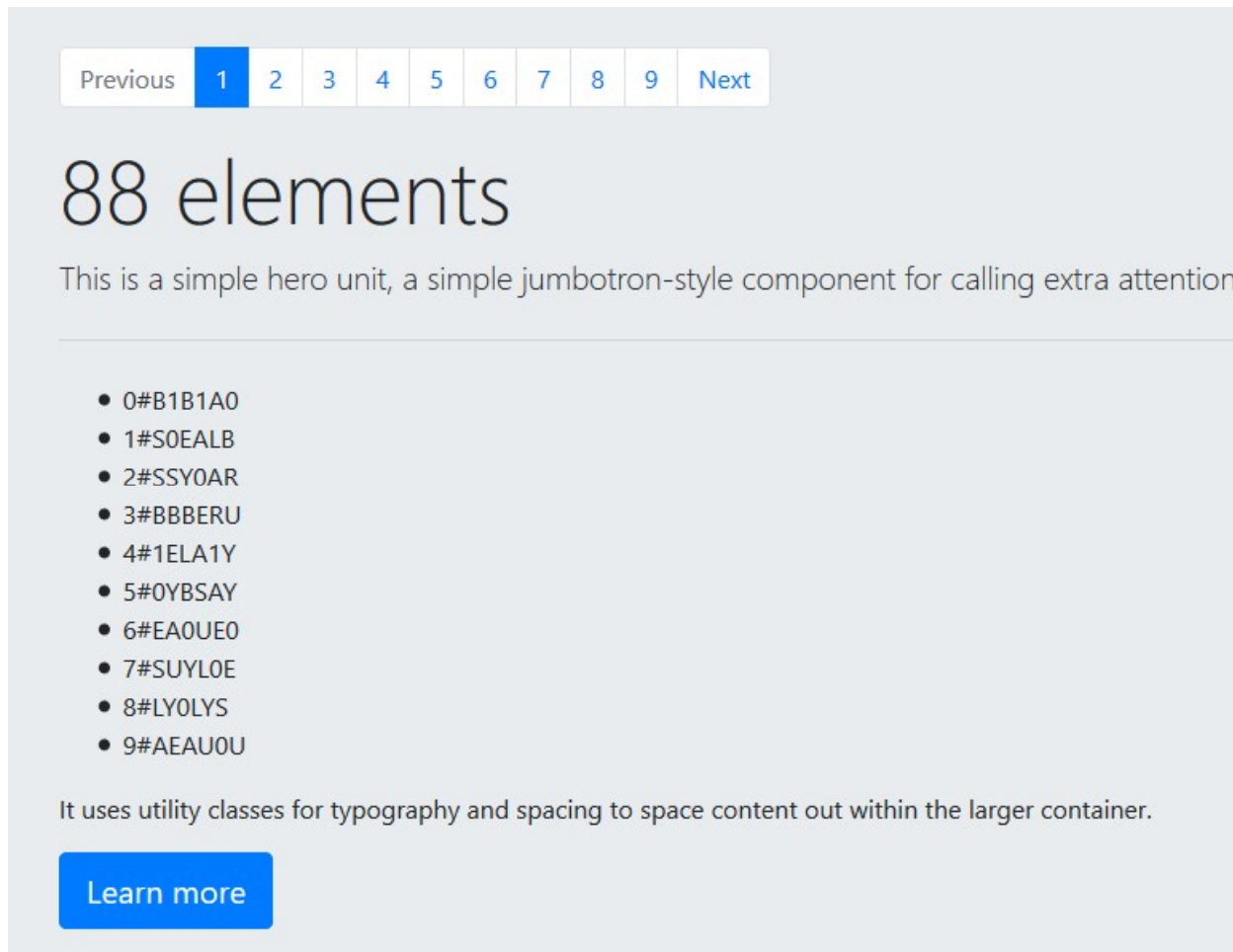
```
this.arrFilter = this.itemArray.slice( beginSlice, beginSlice + 10 )
```



Pasek paginacji zabezpieczony jest przed wybraniem elementu mniejszego niż 1 i większego niż maksymalna liczba elementów a także podświetlany jest wybrany element.

Komponent paginacji jest dzieckiem w stosunku do komponentu strony i w tym rozwiązaniu stosują dekoratory @Input i @Output do przekazywania danych.

3. Opis rozwiązań - Paginacja



4. Prezentacja działania

Navbar

[Home](#) [Do pobrania](#) [Profil](#) [Media](#) [Kontakt](#)



Card title

This is a longer card with supporting text below as a natural lead-in to additional content. This content is a little bit longer.

Last updated 3 mins ago



Card title

This card has supporting text below as a natural lead-in to additional content.

Last updated 3 mins ago



Card title

This is a wider card with supporting text below as a natural lead-in to additional content. This card has even longer content than the first to show that equal height action.

Last updated 3 mins ago

FOOTER CONTENT

LINKS

LINKS

LINKS

5. Podsumowanie oraz wnioski z realizacji projektu

Tworzenie oprogramowania mającego wysoką jakość wymaga od programisty stałego rozwijania umiejętności i uczenie się nowych funkcji a proces ten nigdy się nie kończy.

Framework Angular pozwala na budowanie wydajnych i skalowalnych aplikacji internetowych.

Jest jednym z najpopularniejszych frameworków frontendowych, który pozwala na wygodne tworzenie aplikacji internetowych.

Zaprojektowanie i utworzenie projektu pozwoliło na poznanie działania elementów Angulara takich jak Kontroler, Szablon, Dyrektywa.

6. Plany rozwojowe na przyszłość

Tworzenie aplikacji internetowych dla klientów

Dziękuję za uwagę