Projektowanie i implementacja systemów webowych

Praktyczne informacje na temat Angulara i pracy z modułem frontendu.

AngularCLI

Uruchomienie z katalogu projektowego: `src/main/client`:

```
run.bat ng ...
```

Zarządzanie buildem:

AngularCLI	Skrót	
`run ng build`	`run build`	Zbudowanie części frontendowej.
`run ng test`	`run test`	Uruchomienie testów dla części frontendowej w trybie TDD.
`run ng lint`	`run lint`	Uruchomienie narzędzia lint/tslint do statycznej analizy kodu frontendu.
`run ng serve`	`run start` `run start-proxy`	Uruchomienie frontendu w trybie nasłuchiwania zmian.

UWAGA: backend należy uruchamiać oddzielnie.

Moduły Angular

Moduł jest narzędziem pozwalającym na strukturalizację aplikacji. Każda aplikacja Angular'owa musi składać się z co najmniej jednego modułu. Na poziomie systemu plików moduły są tworzone w podkatalogach źródeł aplikacji.
Generowanie nowego modułu (AngularCLI):

```
ng g module <module name>
```

Konfiguracja modułu odbywa się w dekoratorach (annotacjach) głównej klasy modułu:

```
@NgModule({
  imports: [
    CommonModule,
    FormsModule,
    RouterModule
  ],
  exports: [
  ],
  declarations: [
    BookDetailsComponent
```

```
  ]
})
export class BookMgmtModule {}
```

Imports - deklaruje chęć użycia innego modułu w ramach nowego modułu.
Exports - udostępnia elementy dla innych modułów, które importują nasz moduł.
Declarations - deklaracja elementów wykorzystywanych przez nasz moduł (np komponentów, dyrektyw itp).

UWAGA: aby móc użyć funkcjonalności zawartej w jakimś module, nie wystarczy import klasy lub elementu, konieczne jest także zaimportowanie modułu w ramach dekoratora NgModule.

Komponenty Angular

Komponent jest podstawowym elementem składowym który wykorzystujemy do budowy aplikacji.
Komponent składa się z:

1. **Klasy komponentu** z odpowiednimi dekoratorami - klasa ta pełni funkcję kontrolera w modelu MVC.
2. **Szablonu HTML** odpowiadającego za prezentację komponentu - szablon pełni funkcję widoku w modelu MVC.
3. **Arkusza stylu** CSS/SCSS stylującego szablon (opcjonalnie - style można też (należy) definiować globalnie dla aplikacji).

Generowanie nowego komponentu (AngularCLI):

`ng g component <component name>`

Konfiguracja komponentu także odbywa się przy pomocy dekoratorów klasy komponentu.
UWAGA: AngularCLI nie odpowiada za generowanie modelu (MVC). Model definiujemy przy użyciu klas/interfejsów języka Typescript (Javascript).

Zależności i biblioteki zewnętrzne

Do zarządzania zależnościami od bibliotek zewnętrznych wykorzystujemy narzędzie yarn działające na repozytoriach bibliotek npm. Konfiguracja zależności określona jest w pliku package.json.

UWAGA: Zależności w pliku package.json nie modyfikujemy ręcznie, robimy to przy użyciu komendy yarn.
UWAGA 2: Do wygodnego korzystania z komendy yarn zaleca się wcześniejsze uruchomienie z poziomu terminala skryptu console.bat.

Dodanie nowej zależności:

`yarn add <library name>` - dodanie najnowszej wersji biblioteki
`yarn add <library name>@<version>` - dodanie konkretnej wersji biblioteki

Usunięcie zależności:

`yarn remove <library name>`

Migracja do nowszej (innej) wersji zależności

`yarn upgrade <library name>` - migracja do najnowszej wersji biblioteki
`yarn upgrade <library name>@<version>` - migracja do wskazanej wersji biblioteki

Zależności składowane są w katalogu `node_modules`. Katalog ten nie powinien być składowany w kontroli wersji.
UWAGA: komenda yarn może aktualizować plik `yarn.lock`. Plik ten wraz ze zmodyfikowanym `package.json` przechowujemy w kontroli wersji.