Використання Фреймворку Angular для Розробки Веб-застосунків

Андрій Кочкін
Студент кафедри Інформатики
Харківський національний університет радіоелектроніки
Харків, Україна
andrii.kockin@nure.ua

Діана Руденко
Доцент кафедри Інформатики
Харківський національний університет радіоелектроніки
Харків, Україна
diana.rudenko@nure.ua

Use the Angular Framework for Web Application Development

Andrii Kochkin
Student of Informatics Department
Kharkiv National University of Radio Electronics
Kharkiv, Ukraine
andrii.kockin@nure.ua

Diana Rudenko
Associate Professor of Informatics Department
Kharkiv National University of Radio Electronics
Kharkiv, Ukraine
diana.rudenko@nure.ua

***Анотація*—Angular - написаний на TypeScript front-end фреймворк з відкритим кодом, який розробляється під керівництвом Angular Team у компанії Google, а також спільнотою приватних розробників та корпорацій. Angular — це AngularJS, який переосмислили та який був повністю переписаний тією ж командою розробників.**

***Abstract*—Angular - is a TypeScript-based open-source front-end web application platform led by the Angular Team at Google and by a community of individuals and corporations. Angular is a complete rewrite from the same team that built AngularJS.**

Ключові слова—фреймворк; Angular; компонент; реалізація.

Keywords—Framing; Angular; component; implementation.

I. Порівняння з AngularJS

Основними перевагами Angular є те, що:

- це грамотно і ретельно спроектований, високопродуктивний фреймворк;
- з нижчим порогом входження, порівняно з першою його версією;
- з якісною документацією та великою кількістю практичних прикладів.

Розглянемо Angular у порівнянні з AngularJS - потрібно щоб все придумане лягало на реалії фреймворка, його модулі й сервіси певного типу. Не можна просто взяти і акуратно створити якийсь клас або компонент, який буде робити щось важливе. Потрібно вирішити, чим цей компонент буде з точки зору фреймворка? Яким типом сервісу: value, constant або все ж factory? Може, це буде сервіс типу service? Адже він створює об'єкт з оператором new, здається це те, що потрібно. А раптом Сінглтона буде недостатньо? Такі питання виникають практично завжди під час роботи з AngularJS, і на них немає однозначної відповіді.

Звільнення від подібного роду обмежень, з боку фреймворка є головною перевагою Angular. Можна використовувати будь-яку зручну модульну систему, як завгодно називати і підключати довільний код.

Для початку роботи у AngularJS необхідно:

- створити файлову структуру програми;
- налаштувати роботу з шаблонами;
- налаштувати роботу зі стилями, препроцесором;
- налаштувати збірку для розробки, налаштування, продакшена;
- налаштувати процес тестування.

З другою версією фреймворка ми отримуємо інструмент командного рядка, з якого можна генерувати додатки, модулі, компоненти, директиви, сервіси, фільтри (pipe – нова їх назва), запускати тести, перевірку коду і т. д. І для застосування описаного вище необхідно виконати одну команду:

ng new app- example

Буде створена вся необхідна інфраструктура в кращому на даний момент вигляді. Відразу можна приступати до роботи. Нічого зайвого.

Команда може приймати додаткові аргументи. Наприклад, якщо планується використовувати CSS препроцесор Stylus:

ng new app- example –style = styl

Також будуть автоматично налаштовані компіляція та збирання стилів, з урахуванням обраного препроцесора.

II. Особливості Angular

Згенерований код додатка буде використовувати TypeScript. Насправді це той самий JavaScript (ECMAScript 6), але з деякими приємними і корисними можливостями:

- інтерфейси;
- типізація;
- перерахування (Enum);
- модифікатори (public, private, static);
- декоратори (@).

Все це дозволяє писати більш стабільний і гарний код та звільняє від потреби повсюдно використовувати JSDoc.

В Ангулярі 2 немає контролерів, тільки компоненти. Створити новий можна в такий спосіб:

ng generate component playground/player

Ця команда створить директорію player в playground з мінімально необхідним кодом компонента:

- файл реалізації;
- файл шаблону;
- файл стилів із розширенням використовуваного CSS препроцесора;
- файл юніт-тестів.

Ключова фішка Angular – two-way binding.

<app-player [(position)]=”playerPosition”></app-player>

Такий запис в шаблоні передасть значення властивості playerPosition поточного компонента і буде змінювати його при зміні властивості position усередині компонента player. Це і є two-way binding.

В Angular з’явився новий синтаксис, який дозволяє передавати значення властивостей дочірнім компонентам (one-way binding). Використовує він кутові дужки:

<app-player [position]=”playerPosition”></app-player>

І можна підписуватися на події, що виникають у дочірніх компонентах. Використовуються круглі дужки:

<app-player
(positionChange)=”onPositionChange($event)”></app-player>

Такий запис передбачає, що в компоненті player є властивість positionChange – екземпляр класу EventEmitter. Коли в компоненті player викликається this.positionChange.emit (newValue), виконується код onPositionChange ($event), зазначений у шаблоні. $event буде містити значення newValue.

Так в Angular і реалізується two-way binding:

- передача вихідного значення властивості;
- підписка на подію з назвою «назва властивості всередині дочірнього компонента» + «Change»;
- зміна властивості в батьківському компоненті при появі події.

Завдяки цій реалізації в Ангулярі немає вотчерів, які раніше становили джерела багатьох проблем з продуктивністю.

Angular вирішує питання високого рівня пов’язаності коду, нової потужної реалізації Dependency Injection і можливості абстрагуватися від реалізацій різних взаємопов’язаних компонентів, використовуючи інтерфейси (TypeScript).

Наприклад, можна написати таку конструкцію:

```
class SomeComponent {
constructor (public someService: SomeService) {}
}
```

При створенні екземпляра цього компонента автоматично буде створений екземпляр сервісу SomeService і переданий у конструктор SomeComponent. Це значно знижує рівень пов’язаності й дозволяє тестувати їх окремо один від одного.

Так само запис public someService (TypeScript) робить цей сервіс доступним усередині екземпляра класу за допомогою ключового слова this (this.someService).

Отже, можна сказати, що на даний момент Angular є однім з найкращих фреймворків для розробки WEB-застосунків. Він надає розробнику, який працює з ним, великі можливості у створенні чогось нового.

Література References

[1] Sebastian Eschweiler .“Angular 2” [Online]. https://leanpub.com/angular2-book.

[2] Nate Nurray, Ari Lerner, Carlos Taborda “ng-book. The complete book on Angular”.

[3] Developers are angular, “*official documentation angular*.” https://angular.io/.