PROGRAM SZKOLENIA FRAMEWORKI

# Szkolenie Migracja Angulara

Przeniesiesz projekt na Angulara 22: standalone, sygnały, zoneless i Vitest, bez przepisywania od zera.

| CZAS TRWANIA | POZIOM | FORMA |
| --- | --- | --- |
| **4 dni** | **zaawansowany** | **warsztat dla zespołu, zdalnie lub stacjonarnie** |

Projekt w Angularze sprzed kilku lat działa, ale każda aktualizacja jest coraz trudniejsza: `NgModule` na każdym kroku, Zone.js, Karma i biblioteki, których nikt już nie rozwija. Nowe osoby w zespole uczą się „starego" Angulara, a dokumentacja pokazuje już inny.

Na szkoleniu przeprowadzisz migrację krok po kroku na kodzie podobnym do Twojego: od planu i testów zabezpieczających, przez aktualizację wersji, komponenty standalone i sygnały, po nowe formularze i nowy runner testów. Po każdym etapie aplikacja się buduje, przechodzi testy i nadaje się do wdrożenia. Trzy dni przeprowadzasz migrację aplikacji szkoleniowej, a czwarty poświęcasz na kod Twojego zespołu.

## PROGRAM

**4 dni (3 dni migracji aplikacji szkoleniowej + 1 dzień warsztatu na kodzie Twojego zespołu)**

- Plan migracji
  - Inwentaryzacja: wersje, zależności, biblioteki bez wsparcia
  - Migracja etapami zamiast wielkiego przepisywania
- Testy, które zabezpieczają migrację
  - Co przetestować przed pierwszym `ng update`
  - Testy komponentów, które przetrwają zmianę API
- Aktualizacja wersji i zależności
  - Aktualizacja wersja po wersji z `ng update`
  - Zgodne wersje TypeScriptu, RxJS i Node.js na każdym kroku
  - Przejście na builder `application` (esbuild i Vite)
  - Zastępowanie porzuconych bibliotek
- Zmiany w Angularze 22, które psują build
  - TypeScript 6: `strict` i puste `types` domyślnie, przestarzałe `baseUrl` i `moduleResolution: node`
  - `strictTemplates` domyślnie: błędy typów w szablonach
  - `HttpClient` na Fetch API: interceptory, `reportUploadProgress` i `withXhr()` tam, gdzie potrzebujesz postępu wysyłania

- Komponenty standalone
  - Automatyczna migracja schematem Angular CLI
  - Usuwanie `NgModule` i porządkowanie importów
  - Konfiguracja aplikacji w `app.config.ts`
- Nowa składnia szablonów
  - `@if`, `@for`, `@switch` zamiast `*ngIf` i `*ngFor`
  - Automatyczna migracja schematem `@angular/core:control-flow`
  - `@defer`: leniwe ładowanie fragmentów widoku
- Nowoczesne API frameworka
  - Funkcja `inject()` zamiast wstrzykiwania w konstruktorze
  - `@Service()` zamiast `@Injectable({ providedIn: 'root' })`
  - Funkcyjne guardy, resolvery i interceptory
- Sygnały
  - `signal`, `computed`, `linkedSignal`, `effect`
  - Wejścia i wyjścia jako sygnały (`input()`, `output()`, `model()`)
  - Sygnały a RxJS: `toSignal` i `toObservable`
  - Migracja stanu komponentów krok po kroku
- Formularze i dane asynchroniczne
  - Reactive Forms a Signal Forms: kiedy migrować
  - `resource()` i `httpResource()` zamiast ręcznego `switchMap`
- Wykrywanie zmian
  - OnPush domyślnie: przejście z `Eager` komponent po komponencie
  - Jak działa aplikacja bez Zone.js
  - Przygotowanie kodu i bibliotek do trybu zoneless
- Migracja runnera testów
  - Z Karmy do Vitest: schematy `migrate-karma-to-vitest` i `refactor-jasmine-vitest`
  - `fakeAsync` w Vitest: łatka Zone.js albo fałszywe zegary Vitest
  - Kiedy migracja się opłaca, a kiedy zostać przy Karmie
- Warsztat na kodzie Twojego zespołu (dzień 4)
  - Inwentaryzacja projektu Twojego zespołu i plan migracji etapami
  - Pierwsze kroki migracji w repozytorium Twojego zespołu
  - Angular Material: z komponentów `legacy-*` na MDC i motywy M3 (jeśli projekt go używa)
  - Lista ryzyk i szacunek kosztu kolejnych etapów

Więcej o samych testach znajdziesz na szkoleniu Testy w Angularze.

## DLA KOGO?

- Dla zespołów, które utrzymują projekt w Angularze 12-17 (starsze wersje wymagają dłuższego planu, zapytaj o wariant).
- Dla tech leadów, którzy planują aktualizację i muszą oszacować jej koszt.
- Dla programistów, którzy znają „stary" Angular i chcą pisać nowocześnie.
- Szkolenie nie obejmuje AngularJS (1.x).

## CO MUSISZ UMIEĆ?

- Pracować z Angularem na co dzień: komponenty, serwisy, routing.
- Znać podstawy RxJS i TypeScriptu.

## CZEGO SIĘ NAUCZYSZ?

- Zaplanujesz migrację etapami, bez zatrzymywania prac nad produktem.
- Zabezpieczysz migrację testami przed pierwszą aktualizacją.
- Zaktualizujesz Angulara, TypeScript i RxJS wersja po wersji i naprawisz zmiany z Angulara 22, które psują build.
- Jeśli projekt używa Angular Material, zaczniesz przenosić go z komponentów `legacy-*` na MDC i motywy M3 i oszacujesz resztę tej pracy.
- Przeniesiesz komponenty na standalone i nową składnię szablonów.
- Zastąpisz porzucone biblioteki i przejdziesz na funkcyjne API frameworka.
- Zastąpisz część RxJS sygnałami, `httpResource()` i Signal Forms tam, gdzie to upraszcza kod.
- Przygotujesz aplikację na OnPush i pracę bez Zone.js.
- Przeniesiesz testy z Karmy na Vitest.
- Przygotujesz plan migracji i jej pierwsze kroki na kodzie Twojego zespołu.

## OPINIE UCZESTNIKÓW

> *„Podczas 3-dniowego kursu rozwijaliśmy przykładową aplikację, ćwicząc na niej poszczególne elementy Angulara. Taki sposób nauki był dużo ciekawszy niż wykonywanie osobnych ćwiczeń do poszczególnych zagadnień.”*
>
> Adam, Dyrektor Działu Informatyki (szkolenie z Angulara)

> *„Oprócz samego Angulara Piotr zdążył nam przekazać mnóstwo wiedzy na temat Git, Visual Studio Code, TypeScript i dobrych praktyk programistycznych - czego oczywiście nie było w programie.”*
>
> Sebastian, Full-Stack Developer (szkolenie z Angulara)

## TRENER

### Piotr Kowalski

W branży IT od 2008 roku. Od 2014 jest mentorem dla wielu developerów. Jako trener spędził na sali szkoleniowej ponad 1000 godzin, a na LinkedIn ma 52 rekomendacje. Każde jego szkolenie to dawka solidnej wiedzy oraz humoru.

Więcej: piecioshka.pl/piotr-kowalski

## ZORGANIZUJMY TO SZKOLENIE W TWOIM ZESPOLE

Program dopasuję do Waszego projektu i poziomu grupy. Napisz, z czym pracujecie, a odpowiem z propozycją terminu i zakresu.

| | |
|---|---|
| KONTAKT | **piecioshka.pl/kontakt** |
| E-MAIL | **piecioshka+blog [małpka] gmail.com** |
| SZKOLENIE | **piecioshka.pl/szkolenia/migracja-angulara/** |