BO·TT·EGA
IT minds

Program szkolenia:

Zaawansowane programowanie w TypeScript

Informacje:

Nazwa:	**Zaawansowane programowanie w TypeScript**
Kod:	**JS-TypeScript-adv**
Kategoria:	JavaScript
Grupa docelowa:	architekci developerzy
Czas trwania:	3 dni
Forma:	40% teoria, 40% ćwiczenia, 20% praca w grupie

Szkolenie przeznaczone zarówno dla programistów frontendowych jak i backendowych. Zakres obejmuje zarówno podstawy jak i tematy bardzo zaawansowane.

Szkolenie kładzie nacisk na statyczne typowanie jako alternatywę do dynamicznego JavaScriptu. Uczestnicy poznają zalety płynące z większej kontroli nad typami danych oraz koszty, jakie niesie ze sobą stosowanie TypeScriptu w projektach o różnej skali. Poznają wzorce projektowe stosowane we frontendzie (oraz implementują niektóre z nich). Szkolenie przewiduje także zadania związane z projektowaniem aplikacji, a nie tylko samym kodowaniem.

Zalety szkolenia:

- Zawiera wzorce i najlepsze praktyki
- Poprawne zrozumienie TS w odniesieniu do JS
- Architektura i elementy DDD

BO·TT·EGA
IT minds

Szczegółowy program:

1. Introduction

- 1.1. Chosen elements of ES5 and ES6
- 1.2. Compile-time & runtime
- 1.3. Responsibilities of TypeScript: problems solved & unsolved

2. Types

- 2.1. Primitive types
- 2.2. The any type
- 2.3. Enums, String literals, Tuples
- 2.4. Unions, Intersections, Index types
- 2.5. Function types
  - 2.5.1. Functional Programming with TS

3. Type system

- 3.1. Static vs Dynamic typing
- 3.2. Strong vs Weak typing
- 3.3. Duck typing
- 3.4. Type inference

4. TypeScript Classes

- 4.1. Interfaces
- 4.2. Classes
- 4.3. Mixins
- 4.4. OOP: abstraction, polymorphism, inheritance, encapsulation

5. Ecosystem

- 5.1. Editors/IDEs

BOTTEGA IT minds

- 5.2. Compiler, compilation target
- 5.3. Handling dependencies
  - 5.3.1. .d.ts files
  - 5.3.2. DefinitelyTyped, typings
  - 5.3.3. npm @types
  - 5.3.4. writing custom declarations

6. Advanced Concepts

- 6.1. TS Generics
- 6.2. TS Decorators

7. Bundles

- 7.1. TS Namespaces
- 7.2. TS Modules
- 7.3. Webpack automation

8. TypeScript and legacy code

- 8.1. Project Remake Strategy: one-big- shot vs step-by- step
- 8.2. Moving logic between server & client
- 8.3. Old and new code co-existing
- 8.4. Study case

9. More usecases

- 9.1. Typed templates
- 9.2. Typed promises and other async operations

10. DDD elements

- 10.1. Domain logic in frontend layer
- 10.2. Data Transfer Object
- 10.3. Value Object

BO·TT·EGA
IT minds

11. Contract-First Design (API Contracting)

11.1. TS interfaces contracts

11.2. RAML/swagger-based contracts

11.3. JSON format, JSON Schema

12. Backend-less development

12.1. Organizational and Business background

12.2. BLD implementations