

# Asynchronous JavaScript

*Program szkolenia, Ducin IT Consulting*

Czas trwania: 3 dni

Formuła: 50% wykłady, 50% ćwiczenia

Szkolenie przeznaczone dla osób mających przynajmniej podstawową wiedzę o JavaScriptcie. Poświęcone jest kluczowemu zagadnieniu języka - asynchroniczności, jej obsłudze, projektowaniu przepływu kontroli, skalowaniu. Przydatne jest również dla programistów z backgroundem backendowym, którzy nigdy nie zetknęli się (lub w niewielkim stopniu) z zagadnieniami, które w JavaScriptcie są chlebem powszednim. Po treningu uczestnicy rozumieją analizowane problemy, potrafią je celnie zdiagnozować i dobrać odpowiednie rozwiązanie, zgodnie z tzw. best practices. Potrafią także zaprojektować skalowalne rozwiązania dla (bardzo) dużych aplikacji.

Zakres obejmuje przedstawienie klas typowych problemów asynchroniczności, wzorców rozwiązań, dobór narzędzi i wreszcie implementację.

Podczas szkolenia kładziemy duży nacisk zarówno na zrozumienie istoty omawianych zagadnień, kodowanie własnych rozwiązań, jak i pracę w grupie.

## **Zakres**

W zależności od potrzeb grupy można dostosować materiał do poziomu początkującego lub zaawansowanego

## **Zalety Szkolenia:**

- Fundamenty oraz reguły programowania asynchronicznego w JavaScript
- Problemy asynchroniczności – kompleksowy przegląd rozwiązań
- Klasyczne problemy oraz ich rozwiązania

# Program szkolenia:

## 1. JavaScript Functional Programming

- 1.1. Functions, Function Objects
- 1.2. Contexts
- 1.3. Pure Functions, Side Effects
- 1.4. Scopes: Function vs Lexical
- 1.5. Closures

## 2. Asynchrony

- 2.1. 3 Programming Models: Synchronous, Asynchronous, Parallel
- 2.2. JavaScript inside browsers and node.js
- 2.3. Event Loop, WEB APIs
- 2.4. Run to Completion Rule
- 2.5. Race Conditions
- 2.6. Patterns: Callbacks, Events, Promises, RxJS

## 3. Callbacks

- 3.1. Synchronous & Asynchronous Callbacks
- 3.2. Callback Hell
- 3.3. Events

## 4. Events

- 4.1. Subscriptions
- 4.2. Event bubbling
- 4.3. Event-driven Architectures
- 4.4. Pub-sub: pattern, implementations

## 5. Promises

- 5.1. Promise Design Pattern
- 5.2. States, State Transitions
- 5.3. Chaining
- 5.4. Values
- 5.5. Error Handling
- 5.6. Advanced Patterns & Usecases (Combining Promises)
- 5.7. Promise Anti-patterns

## 6. Promise Implementations

- 6.1. Promises/A+ Specification
- 6.2. ECMAScript 6: Promise, Arrow Functions, Destructuring
- 6.3. Bluebird Promises
- 6.4. jQuery Promises & Deferreds

## 7. Async Await

- 7.1. ES6 generators & ES6 Promises
- 7.2. Couroutines
- 7.3. Async Await
- 7.4. Parallel & Concurrent processing

## 8. RxJS basics

- 8.1. Reactivity basics
- 8.2. Reactive Functional Programming
- 8.3. Observable Pattern
- 8.4. Sequences, Operators, Marble Diagrams
- 8.5. Error Handling
- 8.6. Subjects
- 8.7. Observables vs Promises: differences & similarities