Modeling modern systems in the context of the Spring Framework

Take a step towards becoming a **Technical Partner for Business** and learn how to use the power of the most popular Java framework in the context of creating applications of the highest quality.

Workshop training during which you and your trainer will build a well-designed application using DDD, TDD and Clean Architecture practices. The heart of the application will be integrated with the Spring Framework only in the necessary places, thus maintaining the purity of the architecture and domain modules.

This training is not intended to cover all Spring Framework mechanisms and modules. Instead, you'll get hands-on guidance and practice techniques to build top-notch, easy-to-modify apps. Note: Spring will only appear when we deem it necessary.

The training consists of two parts:

1. The heart of the application - universal techniques for creating good software with an implementation detail in the form of Spring Framework
    a. Techniki Knowledge Crunching
    b. DDD
    c. TDD
    d. Selection of architecture
    e. Modular monolith
2. Deep dive into Spring Framework
    a. Operation details and internale
    b. Traps
    c. Best practices
    d. Integration tests
    e. Spring Framework projects

Program

Part 1 - the heart of the application

1. Introduction to a sample domain
2. Knowledge crunching
    a. Product vision
    b. Big Picture
    c. Discovery of critical processes
    d. Example mapping

3. Domain code modeling using TDD in the mob programming formula - Code Storming
4. Architecture selection rules
5. An example of using the Ports & Adapters architecture
6. Integration tests
7. Preparation of the application as a modular monolith
    a. Selection of architecture for the module
    b. Communication patterns between modules
8. Introduction to the Spring Framework as an implementation detail of selected modules
9. Organization of modules based on the Spring Framework

Część 2 - Deep dive into Spring Framework

1. Container
    a. Let's make your own container (optional)
    b. Configuration
    c. Container implementations
    d. hierarchy
    e. Principle of operation - Inversion of Control (in particular Dependency Injection)
2. Components
    a. Declaration - Java config and annotations
    b. Life cycle
    c. Dependence
    d. Component range
        i. Scoped Proxy traps
        ii. Injecting shorter-lived components into longer-lived ones
    e. Pitfalls of annotations (e.g. @Transactional)
    f. Pre/post processors
3. AoP
4. Spring Data
    a. JPA
    b. MongoDb
    c. JDBC
5. Creating a real RESTful API according to the Richardson Maturity Model
6. Spring Security
7. Support for asynchronous programming
8. Support for event-driven programming
9. Spring Boot
10. Introduction to Spring Cloud