IPT – Intellectual Products & Technologies Ltd.
Bl. 44- Office 2 Slatinska Str., Sofia, Bulgaria
☏ (+359-2) 973-38-38 🖷 (+359-2) 971-28-50 ✉ office@iproduct.org 🖰 http://www.iproduct.org/

Introduction to Spring Framework: Hibernate, Spring MVC & REST

Training domain:	**Software Engineering**
Number of modules:	**1**
Duration of the training:	**40 hours**

Sofia, 2017

IPT – Intellectual Products & Technologies Ltd.
Bl. 44- Office 2 Slatinska Str., Sofia, Bulgaria
☏ (+359-2) 973-38-38 🖷 (+359-2) 971-28-50 ✉ office@iproduct.org 🖰 http://www.iproduct.org/

Introduction to Spring Framework: Hibernate, Spring MVC & REST

STUDY PLAN

Module name	Introduction to Spring Framework: Hibernate, Spring MVC & REST
Presentations and demonstrations, hours	20
Hands-on exercises, hours	20
Total, hours	40

Instructor: **Trayan Iliev**

IPT – Intellectual Products & Technologies Ltd.
E-mail: tiliev@iproduct.org

Target audience: Java SE/ Web developers

Duration: The total duration of the training is 40 hours.

Key takeaways:

- Embrace the rich opportunities for rapid web and SOA application development with Spring 5 ecosystem of projects and modules;
- Develop, deploy, optimize, secure, and test production grade web applications, (micro-) services and clients with *Hibernate, Spring MVC & REST;*
- Plenty of hands-on experience with *Spring Boot*, *Hibernate, JPA, Spring MVC, Spring Data, Spring Security, Spring MVC Test framework*, *JUnit*, *Mockito, Selenium* / *FluentLenium;*
- Novelties in *Spring 5 WebFlux* & *Spring Boot 2.0* – functional reactive programming and security model, reactive event streaming.

Training description:

The training introduces to enterprise SOA & web application development using Spring 5 platform. The main topics include:

1. **Introduction to Spring** – evolution of *Spring framework*, main features, *Spring* main modules. Introduction to *Maven* and *Gradle*. Building a *HelloSpring* application using XML and annotation-based configurations. Introduction to *Spring Boot* – building *HelloSpringMVC* simple web application using *spring-boot-starter-*

web. Configuring dispatcher, resolver, static resources, locale, multipart, error handling and encoding with *Spring Boot* and *Spring* embedded servlet container (*Tomcat*). (2 h)

2. **Inversion of Control (IoC) and Dependency Injection (DI) in Spring** – lookup vs. injection, constructor, setter and field-based DI. Instantiating the container. *Beans* and *BeanFactory* implementations. Configuring *ApplicationContext* – basic configuration, classpath scanning and filters, component declaring meta-annotations. XML-based configuration using *GenericXmlApplicationContext*. Java annotations configuration (*@Bean*, *@Configuration*, *@ComponentScan*) - *AnnotationConfigApplicationContext*. Mixing XML & Java - *@Import*, *@ImportResource*. Instantiating beans using constructor and static/instance factory methods. Dependency resolution process. Dependencies and configuration in detail – values, bean references, inner beans, collections, maps, *null* handling, *p*- and *c*-namespaces, compound property names, *depends-on*, lazy initialization, autowiring. Excluding a bean from autowiring. Limitations and disadvantages of autowiring. (4 h)
3. **Introduction to Spring Web MVC** – *Spring MVC* architecture: *DispatcherServlet, HandlerMappingm Controller, ViewResolver, View*. Using *Thymeleaf* templates to build our first page. Building simple controller - *@Controller, @RequestMapping*. Passing data to the View using *Model, ModelMap*, and *ModelAndView*. Getting data with a request parameter – *@RequestParam*. Building a sample Spring MVC application. (2 h)
4. **Inversion of Control (IoC) and Dependency Injection (DI) in Spring (Advanced)** – Bean scopes - *singleton, request, session, application*, *websocket*, and custom scopes. Web configuration. Scoped beans as dependencies - *<aop:scoped-proxy>*. Lifecycle callbacks – initialization, destruction, *@PostConstruct*, *@PreDestroy*. *ApplicationContextAware*, *BeanNameAware* interfaces. Using Spring resource injection annotations - *@Autowired, @Component, @Scope, @Value, @Required, @Qualifier, @Lazy, @Primary.* Using JSR 330 standard annotations - *@Resource, @Inject, @Named, @ManagedBean, @Qualifier, @Alternative.* Environment and profiles, *@PropertySource*. I18N using *MessageSource*. Events and event listeners - *@EventListener*, *@Order*, *@Async*. (3 h)
5. **Resources and resource loaders. Validation, data binding, and type conversion.** *JSR-349 Bean Validation 1.1*. Spring's *Validator* interface. Resolving codes to error messages. *PropertyEditors* and type conversion. Null-safety annotations. (1 h)

6. **Handling forms and complex URL mapping** – validation, customizing validation messages. Enabling internationalization (*i18n*) and localization (*l10n*) – configuring *i18n* in the *DispatcherServlet* configuration, changing locale, translating application text. Displaying lists. Request mapping in depth – HTTP methods, URI patterns, suffix match, matrix variables, parameters, headers. Handling file uploads. Handling errors and error message translation. Custom error pages. Using sessions. (3 h)
7. **Introduction to Spring Expression Language (SpEL) and Aspect Oriented Programming (AOP) with Spring**. AOP concepts – *aspect, advice, joint point, pointcut, introduction, target object, AOP proxy*. Types of *advices*. Dynamic proxies. Declaring *pointcuts. Advisors* examples: declarative transactions, retrying operation execution. (2 h)
8. **Spring JDBC Support** – DAO pattern, simple relational DB example, Spring JDBC Infrastructure, database Connections and *DataSources*, embedded database support, exception handling. *JdbcTemplate* class. Retrieving nested entities with *ResultSetExtractor*. Spring classes modelling main JDBC operations. Inserting data and retrieving the generated key. Spring Data project - JDBC extensions. *Spring Boot* starter JDBC. (3 h)
9. **Object to Relational Mapping (ORM). Using Hibernate and JPA with Spring. Transactions** – *Hibernate API* vs. *Java Persistence API (JPA)*. Important ORM concepts using Hibernate. Configuring the Hibernate *SessionFactory*. ORM mapping using Hibernate annotations. Hibernate *Session* interface – querying, inserting, updating and deleting data. Annotating fields or properties. *Spring Data* access using *JPA 2.1* – ORM mapping using JPA annotations, database operations using *JPA* Repository abstraction, custom queries. *Spring Boot* starter data *JPA*. Validation using *Beans Validation* annotations. Considerations choosing the right data management approach. (4 h)
10. **Transaction management** – transaction types, interfaces *PlatformTransactionManager*, *TransactionDefinition* and *TransactionStatus*, AOP-based transaction management config. Using programmatic transactions. (1 h)
11. **Building RESTful Web Services with Spring** – introduction to SOA, REST and HATEOAS, levels of maturity of web applications, resources and sub-resources, URI templates. Using Spring MVC to expose RESTful web services – *@Controller, @RequestMapping, @ResponseBody, @RestController*. Configuring JSON/XML data binding. Configuring *Spring* web application. Using curl client to test the REST service. Building dedicated Spring client using *RestTemplate*. Securing RESTful web service with *Spring Security* – *@EnableWebSecurity,*

WebSecurityConfigurerAdapter, HttpSecurity, *AuthenticationManagerBuilder.* Documentation with *Swagger*. RESTful web services with *Spring Boot*. (3 h)

12. **Building fully featured web applications with Spring Web MVC** – implementing multi-layer web applications including DAO, service and presentation layers. Spring MVC architecture – requests life-cycle, responding to normal HTTP and Ajax requests, *WebApplicationContext* hierarchy, main components – handlers, handler interceptors, error and view resolvers, configuration. Spring integration with different view technologies. Implementing views with *Thymeleaf*. Securing web apps with *Spring Security*. *Cross Origin Resource Sharing (CORS)*. Production grade Spring MVC optimizations – gzipping, cache control, using application cache *(@EnableCaching, CacheManager, @Cacheable, @CachEvict, @CachePut, @Caching, @CacheConfig*), async methods, *ETags (ShallowEtagHeaderFilter)*. Using *WebSocket API* with *Spring* and *SockJS*. (4 h)
13. **Spring web application testing** – TDD, unit, integration, front-end (end-to-end), and performance testing. Mocks and stubs. Using *Spring Testing* annotations. Unit testing *Spring* components using *JUnit* and *Mockito*. Integration testing – configuring service-layer testing profile, *Spring MVC Test framework*. End-to-end testing with Selenium / FluentLenium.(4 h)
14. **Novelties in Spring 5: WebFlux** – reactive programming, Reactive Streams Pproject Reactor. REST services with *WebFlux* –annotation-based and functional reactive. Router, handler and filter functions. Reactive repositories and database access with *Spring Data*. End-to-end non-blocking reactive web services with *Netty*. Reactive *WebClients* and integration testing. Reactive *WebSocket* support. Real-time event streaming to *JS* clients using *SSE*. (4 h)

Presentations, demonstrations, and hands-on exercises are conducted in parallel to achieve immediate reinforcement of concepts in practice.

Participants will learn how to develop, deploy, and test production grade, secure web apps and services with *Spring Boot, Hibernate, Spring MVC, WebFlux and REST*. Study is conducted in small groups – up to 10 persons. During the training there will be opportunity for discussion of additional questions the participants are interested in.