**IT355-DZ15**

 **What is Dependency Injection?**

Dependency Injection (DI) je dizajn šablon koji se koristi za implementaciju Inversion of Control (IoC), omogućavajući klasi da dobije svoje zavisnosti iz eksternog izvora umesto da ih sama kreira. Ovo promoviše slabu povezanost i poboljšava testabilnost i održivost koda.

 **Why is Spring one of the most popular Java related frameworks?**

Spring je popularan zbog sveobuhvatne infrastrukturne podrške za razvoj Java aplikacija. Pruža širok spektar funkcionalnosti kao što su dependency injection, upravljanje transakcijama, web MVC okvir, AOP, integracija sa drugim okvirima i podrška za RESTful servise, između ostalog. Njegova modularna priroda omogućava programerima da koriste ono što im je potrebno i izostave ono što im nije, što ga čini laganim i fleksibilnim.

 **What are the different modules in Spring Framework?**

Ključni moduli u Spring Framework-u uključuju:

* Core Container: Core, Beans, Context, i Expression Language.
* Data Access/Integration: JDBC, ORM, OXM, JMS, i Transactions.
* Web: Web, Web-Servlet (Spring MVC), Web-Socket, i Web-Portlet.
* AOP (Aspect-Oriented Programming): AOP i Aspects.
* Instrumentation: Instrumentation i Aspects.
* Test: Test modul.

 **Can you give an overview of a web application that is implemented using Spring related Modules?**

Tipična web aplikacija zasnovana na Spring-u bi uključivala:

* **Spring MVC** za rukovanje web zahtevima i odgovorima.
* **Spring Boot** za pojednostavljivanje postavke i konfiguracije aplikacije.
* **Spring Data JPA** za pristup podacima i ORM (Object Relational Mapping).
* **Spring Security** za autentifikaciju i autorizaciju.
* **Spring Rest** za kreiranje RESTful web servisa.
* **Spring AOP** za rukovanje aspektima kao što su logovanje i upravljanje transakcijama.
* **Spring Batch** za rukovanje batch procesiranjem

 **What is the simplest way of ensuring that we are using single version of all Spring related dependencies?**

Korišćenje alata za upravljanje zavisnostima kao što su Maven ili Gradle sa "Spring BOM (Bill of Materials)" osigurava da sve Spring zavisnosti budu iste verzije. Za Maven, možeš koristiti spring-boot-dependencies BOM u svom pom.xml fajlu.

 **What are the major features in different versions of Spring?**

Glavne funkcionalnosti uključuju:

* **Spring 2.0**: Poboljšan AOP, nova XML Schema-based konfiguracija.
* **Spring 3.0**: Podrška za Java 5, expression language (SpEL), REST podrška.
* **Spring 4.0**: Podrška za Java 8, Groovy bean definition DSL, WebSocket podrška.
* **Spring 5.0**: Podrška za reaktivno programiranje, funkcionalni web framework, podrška za Java 9.
* **Spring 5.2**: Podrška za Kotlin koroutine, RSocket integracija.

 **What are the latest specifications supported by Spring 4.0?**

Spring 4.0 podržava:

* Java 8 funkcionalnosti.
* Java EE 6 i 7.
* WebSocket.
* Poboljšanja za REST.
* Groovy bean definition DSL.

 **Can you describe some of the new features in Spring 4.0?**

Nove funkcionalnosti u Spring 4.0 uključuju:

* Potpuna podrška za Java 8.
* Groovy DSL za definicije bean-ova.
* Podrška za WebSocket i SockJS.
* Opšta poboljšanja u core kontejneru, tip-safe injekcija i podrška za Java EE 7.

 **What is auto-wiring?**

Auto-wiring je funkcionalnost u Spring-u koja omogućava Spring-u da automatski rešava i injektuje bean-ove saradnike u bean. Ovo može biti urađeno korišćenjem različitih modova kao što su byType, byName, constructor i autodetect.

 **What would happen in Spring Container finds multiple bean definitions matching the property to be auto wired?**

Ako više bean-ova odgovara, a nema primarnog bean-a ili specificiranog kvalifikatora, Spring kontejner će baciti izuzetak zbog nejasnoće.

 **How is Spring’s singleton bean different from Gang of Four Singleton Pattern?**

Spring-ov singleton scope osigurava da je samo jedna instanca bean-a kreirana po Spring IoC kontejneru, dok Gang of Four Singleton Pattern osigurava jednu instancu po classloader-u ili JVM-u. Spring-ov singleton scope se takođe može lako konfigurisati i podržava dependency injection.

 **How do you represent stateful bean in Spring?**

Stateful bean-ovi u Spring-u mogu biti predstavljeni korišćenjem @Scope anotacije sa ConfigurableBeanFactory.SCOPE\_PROTOTYPE ili @Scope("prototype"). Ovo osigurava da se nova instanca bean-a kreira svaki put kada se zatraži.

 **How do you use values defined in a property file in an application context xml?**

Možeš koristiti PropertyPlaceholderConfigurer ili PropertySourcesPlaceholderConfigurer za učitavanje property fajlova. Koristi ${propertyName} za referenciranje vrednosti property-a u application context XML-u.

 **How is validation done using Spring Framework?**

Validacija se može uraditi korišćenjem Spring-ovog Validator interfejsa ili JSR-303/JSR-380 Bean Validation API-ja sa @Valid i @Validated anotacijama u kombinaciji sa validator implementacijama kao što je Hibernate Validator.

 **How do you implement cross cutting concerns in a web application?**

Cross-cutting concerns kao što su logovanje, upravljanje transakcijama i sigurnost mogu se implementirati korišćenjem Spring AOP (Aspect-Oriented Programming) ili ugrađenih funkcionalnosti Spring Boot-a kao što su @Transactional i @Aspect.

 **What is an Aspect and Pointcut in AOP?**

Aspect je modul koji enkapsulira cross-cutting concerns, dok je pointcut predikat koji odgovara join point-ovima (tačkama u izvršenju programa kao što su izvršenja metoda) kako bi se odredilo gde treba primeniti advice (akciju koju preduzima aspect).

 **What are the different types of AOP advices?**

Različite vrste AOP advices uključuju:

* Before advice: Izvršava se pre join point-a.
* After advice: Izvršava se nakon join point-a.
* After returning advice: Izvršava se nakon što join point završi normalno.
* After throwing advice: Izvršava se ako metoda izlazi bacanjem izuzetka.
* Around advice: Izvršava se pre i nakon join point-a.

 **How do you define transaction management for Spring – Hibernate integration?**

Upravljanje transakcijama može se definisati korišćenjem Spring-ove @Transactional anotacije ili XML konfiguracije. Mogu se specificirati atributi transakcije i pointcut-ove za upravljanje transakcijama deklarativno.

 **How do you choose the framework to implement AOP - Spring AOP or AspectJ?**

Spring AOP je dovoljan za osnovne potrebe AOP-a i lakši je za korišćenje unutar Spring aplikacije. AspectJ je moćniji i fleksibilniji, podržava složenije pointcut izraze i mehanizme tkanja, ali zahteva dodatnu postavku.

 **What are the different mock objects provided by Spring test framework?**

Spring pruža nekoliko mock objekata uključujući MockHttpServletRequest, MockHttpServletResponse, MockHttpSession, MockServletContext, i MockMvc za testiranje web aplikacija.

 **What are the utility methods available to test JDBC classes?**

Spring-ova JdbcTestUtils klasa pruža utility metode za testiranje JDBC klasa, kao što su countRowsInTable, deleteFromTable, i executeSqlScript.

 **How do you setup a Session Factory to integrate Spring and Hibernate?**

SessionFactory se može postaviti konfigurišući LocalSessionFactoryBean u svojoj Spring konfiguraciji. Treba se obezbediti datasource osobine, Hibernate osobine i anotirane klase ili pakete LocalSessionFactoryBean.

 **How do you implement caching with Spring framework?**

Keširanje u Spring-u može se implementirati korišćenjem anotacija @Cacheable, @CachePut, i @CacheEvict. Treba se konfigurisati cache manager (kao što su EhCache, Redis ili Caffeine) u Spring konfiguraciji.

 **What are the important features of Spring Batch?**

Važne funkcionalnosti Spring Batch-a uključuju:

* Deklarativni I/O (ItemReader, ItemWriter).
* Upravljanje transakcijama.
* Chunk processing.
* Particionisanje poslova.
* Job i step listener-i.
* Retry i skip logika.
* Podrška za različite izvore podataka.

 **What are the important concepts related to setting up a Job in Spring Batch?**

Važni koncepti uključuju:

* Job: Predstavlja batch job.
* Step: Faza u batch job-u.
* JobInstance: Predstavlja jedno pokretanje posla.
* JobExecution: Predstavlja izvršenje JobInstance.
* StepExecution: Predstavlja izvršenje koraka.
* ItemReader: Čita stavke.
* ItemProcessor: Procesira stavke.
* ItemWriter: Piše stavke.

 **What are the different ItemReader and ItemWriter implementations available with Spring Batch?**

Spring Batch pruža nekoliko implementacija, uključujući:

* ItemReader: FlatFileItemReader, JdbcCursorItemReader, JpaPagingItemReader.
* ItemWriter: FlatFileItemWriter, JdbcBatchItemWriter, JpaItemWriter.

 **How do you start running a Spring Batch Job?**

Spring Batch job može se pokrenuti korišćenjem JobLauncher. Može se pokrenuti programski, putem web interfejsa ili korišćenjem scheduler-a kao što je Quartz.

 **How do you configure parallel execution of steps with Spring Batch?**

Paralelno izvršenje može se konfigurisati korišćenjem TaskExecutor u konfiguraciji koraka. Može se definisati TaskExecutor bean kao što su SimpleAsyncTaskExecutor ili ThreadPoolTaskExecutor i konfigurisati korak da ga koristi.