PHASE 3 TOPICS

**SPRING CORE**

**20 March 2023 – DAY 1**

1. Spring is basically built on 3 core principles
   1. AOP – Aspect Oriented Programming  
      is used to provide common functionalities within your spring application context
   2. DI – dependency Injection  
      is used to inject any dependencies that your application may have
   3. Abstraction layer over the existing API’s
2. Create a spring quickstart maven project
   1. Add dependency in pom.xml file spring-context
3. Also create a core java project and
   1. add dependencies downloading from  
      <https://repo.spring.io/ui/native/release/org/springframework/spring/>
   2. Add dependencies in the build path
4. Created our java classes -> every java class in spring is called as a bean
5. Create spring.xml => spring configuration file to add spring related configurations
   1. If maven create within src/main/resources folder
   2. If normal java project then add in src folder
6. Dependency injection
   1. XML based approach
      1. Configure java classes using <bean class=”fully qualified clas name”> . id is optional but it is mandatory when there is ambiguity of the beans i.e more than 1 instance of same bean is configured
      2. Constructor injection  
         <constructor-arg>
      3. Property / setter injection  
         <property>
      4. Scoping in spring
         1. Singleton : which is default as only 1 instance per application is created
         2. Prototype : which creates different instances everytime getBean is called
         3. Request and session used in web applicaitons
   2. Annotation based approach
      1. @Component => is added on the class name to tell spring that this bean or class needs to be loaded and instantiated.
      2. @Value => used for injecting primitive type dependencies
      3. @Autowired => used for injecting reference type dependencies
      4. @Configuration => used as an alternative for xml based configuration. It is used on the class
      5. @ComponentScan => will tell spring to look up for all the beans with spring annotations [ Component, Repository, Service, Controller, RestController, Bean ] and load them. It default looks in the current / root package and the sub packages
      6. @Primary => can be used on the class to be primary bean injected for ambiguity
      7. @Qualifier => can be used to override the @Primary
7. Load the configuration file , xml  
   ClassPathXmlApplicaitonContext(“spring.xml”)
8. Load the configuration file , annotation  
   AnnotationConfigApplicationContext(AppConfig.class)
9. BeanFactory vs ApplicationContext
10. Event Handling in spring
    1. Create an event of type ApplicationEvent
    2. Create a listener using @EventListener
    3. Create a Publisher using ApplicationPublisherEvent

**21 March 2023 – DAY 2**

1. Collections Mapping
   1. @Autowired
   2. @Bean : can be used on any method within the configuration file.

**SPRING JDBC**

1. Create a quickstart maven project
2. Added context, jdbc and mysql dependencies
3. Create beans for which database table and columns will be mapped
4. Database connection parameters:
   1. Provides an abstraction layer over JDBC API
   2. JdbcTemplate provides with skeleton to connect and execute queries for database
      1. Reference of type DataSource : DriverManagerDataSource [ connection parameters ]
      2. Inject JdbcTemplate with DataSource reference
5. Inject JdbcTemplate in any DAO layer which needs to execute queries for DB
6. Insert / update / delete => jdbcTemplate.update(sql, …)
7. Select by id => queryForObject(sql, RowMapper, ..)
8. Select all => query (sql, RowMapper)
9. Configuration for DB connection parameters can be externalize as follows:
   1. Create properties file under src/main/resources folder and add the db connection parameters as key=value pairs
   2. Use @PropertySource to load the properties file
10. **PASSWORDS SHOULD BE ENCRYPTED - JASYPT**  
    <https://www.baeldung.com/spring-boot-jasypt>