Spring Tutorial Notes

xml should be placed outside the com packages

notes:

- **3 ways of config:**
  - **full xml**
  - **xml component**
    - **<context: component - scan base-package=“name of the package” />**
  - **java config**

Annotation: (no need to use xml to define an object)

1. @Component — on top of object class
2. In client code, use lowercased version of the object class name

or

1. @Component(“name”)
2. In client code, specify the object using “name”

Autowiring: (use method without specifying, but look for it in components)

(constructor injection, setter injection, field injection)

- **Constructor injection:**

1. @Component — on top of the object class where method desired is defined
2. @Autowired — on top of the constructor where we use unspecified method

- **Setter injection: (inject dependencies by calling setter methods on class)**

1. same but in setter

as a matter of fact, any method can use tech

- **Field injection: (injection dependencies by setting field values on your class directly)**

1. same but on top of field

- **autowiring and qualifiers**

1. @Qualifier(class name with lowercase) under @Autowired

Bean scope with annotations

- @Scope(“desired_scope”) — on top of the object class. (singleton,

prototype, etc.)

Bean life cycle

- @PostConstruct (will execute after constructor is called)

and

- @PreDestroy (will execute before bean is destroy)

on methods

Spring config using Java code (no xml)

- **General define process**

1. Create a java class and annotated as @Configuration
2. Add component scanning support: @ComponentScan(“package_name”)
3. Read spring java configuration class **(AnnotationConfigApplicationContext context = new AnnotationConfigApplicationContext(SportConfig.class);**
4. Retrieve bean from spring container **(same as xml version)**

- **Define beans in spring**

1. Define method to expose bean (@Bean — **on top of method**)
2. Inject bean dependencies
3. Read spring java config class
4. Retrieve bean from spring container

- **Java config props**

1. Create properties file
2. Load properties file in spring config [e.g. @PropertySource("classpath:sport.properties”)]
3. Reference values from properties file [e.g. @Value("${foo.email}”)]

Spring MVC

framework for developing web application

- configuration:

1. Add config to file: WEB-INF/web.xml
    1. Configure spring MVC dispatcher servlet
    2. Set up URL mappings to spring mvc dispatcher servlet
2. Add config to file: WEB-INF/spring-mvc-demo-servlet.xml
    1. Add support for spring component scanning
    2. Add support for conversion, formatting and validation
    3. Configure spring mvc view resolver