Advanced Java

1. Explain Servlet & JSP life cycle?
    - Servet Life Cycle -- Servlet interface
        - init()
        - service() -- per request
        - destroy()
    - Jsp Life Cycle
        - JSP Engine
            - translation .jsp --> .java
            - compilation .java --> .class (servlet class)
        - Servlet Engine
            - initialization --> jspInit()
            - request handling --> _jspService()
            - destruction --> jspDestroy()
2. What is difference between Redirect and RequestDispatcher scenario?
    - Redirect
        - Two requests from client -- due to temp response 302
        - Client is aware of redirection
        - resp.sendRedirect(url);
        - Spring MVC: return "redirect:url"
    - RequestDispatcher
        - Single request from client
            - request forward
            - request include
        - req.getRequestDispatcher(url) --> forward() or include()
        - Spring MVC:return "forward:url"
3. What is session tracking? How to do it in java?
    - HttpSession belongs to which user?
        - Cookie (default -- jsessionid)
        - URL rewriting
4. What is hibernate? Explain hibernate architecture.
    - Java <--> ORM <--> RDBMS
    - Architecture
        - Session and SessionFactory
        - Transaction
        - Dialect -- Lowest layer that produce SQL as per RDBMS.
5. Explain hibernate entity life cycle?
    - Life cycle: Transient, Persistent, Detached and Removed.
    - Session cache --> Persistent --> State is tracked and updated by Hibernate.
    - Transient vs Detached
    - Removed --> Corresponding row is deleted from RDBMS.
6. What is the difference between get() and load() of hibernate?
    - get()
        - SQL query execute, get data, put in entity object and return it.

        - if not found, return null.
        - eager fetch
    - load()
        - get the id, put in wrapper proxy and return it.
        - when object fields are accessed, proxy internally fetch data from RDBMS
        - if not found, throws excetion.
        - lazy fetch
    - find()
        - same as get()
        - JPA compliant
7. How can we call stored procedure in hibernate?
    - @Procedure
    - @NamedStoredProcedure
8. What is IOC and Dependancy injection?
    - Inversion of Control -- Object creation and initialization done by Spring container.
    - Spring container initialize dependencies externally -- setter based, constructor based and field based.
9. What is auto-wiring? Which are the types of auto wiring? What if prototype bean is auto-wired in a singleton bean?
    - Automatically initializing appropriate beans as dependencies into dependent beans.
    - Implicit DI
    - Setter based, * Constructor based and Field based.
    - Prototype into Singleton --> ApplicationContext or @Lookup if need separate prototype bean
10. Explain spring bean life cycle. Explain spring bean scopes.
    - InitializingBean
    - Aware interfaces
    - @PostConstruct
    - BeanPostProcessor
    - @PreDetstroy
    - DisposableBean
11. What is use of @Transactional? Why it should be used on service layer?
    - @Transactional -- AOP style transaction management of Spring.
        - Auto done by TransactionManager.
    - One business logic will need multiple DAOs.
12. Explain Spring MVC life cycle?
    - Refer slide
    - View --> Request --> Front Controller --> HandlerMapping & HandlerAdapater --> User defined controller's request handling method --> Front Controller (view Resolver) --> View
13. What is the difference between SOAP and REST? What is significance of RestController?
    - REST protocol = HTTP protocol + Object state transfer.
        - GET, POST, PUT and DELETE.
        - Stateless.
    - @RestController = @Controller + @ResponseBody (for request handler method return type).
14. What is Spring Boot? What do you mean by opinionated defaults? How auto-configuration works?
    - Spring Boot = Spring framework + Embedded Web Server + Auto config - XML config - Jar conflicts.

- Opinionated defaults = Most commonly used frameworks e.g. Hibernate, Tomcat, ...
- Auto-configuration = Conditional config based on pom.xml dependencies and created beans.
  - Auto-configuration is done by pre-defined @Configuration classes e.g. DataSourceAutoConfiguration, etc.
- Spring starter = Predefined set of dependencies and config to be added into Spring Boot project.
  - Given in terms of Maven dependencies.
  - Aggregate dependency for other small dependency.
  - Example: spring-boot-starter-web = Spring Web MVC + Spring Boot + Jackson + Embedded Tomcat

15. What is significance of @CrossOrigin? How it works?
    - @CrossOrigin -- CORS filter -- Response headers "access-control-allow-origin": "*"
    - Another web application can also access REST API (e.g. React app).
16. State Management
    - Server side
      - HttpSession --> Session tracking
      - ServletContext
      - Request
    - Client side
      - Cookie
      - QueryString
      - HiddenFields
17. Spring Boot -- State Management?
    - Spring MVC
      - Same as above -- Server side + Client side
      - Request handler method -- HttpSession, @Cookie, Model, beans @Scope("request|session|application"), ...
    - REST api
      - REST services are stateless -- no data is stored on server side.
      - Client may store data in -- LocalStorage and SessionStorage.
18. @SpringBootApplication = @Configuration + @ComponentScan (in current package) + @EnableAutoConfiguration.
    - @Configuration -- bean creation, property sources, ...
    - @ComponentScan -- stereo-type annotations
    - @EnableAutoConfiguration -- auto config based in jars in classpath and created beans.