Thursday, July 21, 2022 9:06 AM

What is	Inner class	Within the { } of outer class WE define another class outer class , inner class Types --- simple inner class -- access static+non static members of outer class static inner class -- access only static members of outer class method local inner class --- access only final local variables of the method anonymous inner class ---- ……
	Lambda function	It is a shorthand notation for anonymous inner class
	Reflection	Introspecting the class/object at runtime
	Distributed n-tier	
	JDBC	API to write Java db clients
	Http	protocol , full form, for a web application Http Request Http Request Header , Http Request Body Http Response Http ResponseHeader , Http Response Body
	Web Server	Server-continuously runs usually on 80 / 8080 port , it understands http request and response ex tomcat, express, IIS
	JEE compliant web server	Tomcat - Servlet Container , JSP Engine
	Servlet	WEB component , written in Java with embedded HTML , Embedded in Web server
	JSP	Web component written in html with Embedded Java code
	Deployment	Submit our web component to web server (WAR FILE)
	Hibernate	ORM for Java , full form
	JPA	Wrapper over ORMs to give a common interface to access ORMs
	Spring Framework	Container that manages components called as beans
	Spring Boot	Easy pre configured(ready made tomcat server is embedded, Maven support , dependencies , directory structure) templates that internally uses spring framework
	MVC	architecture -- model , view ,controller (EXAMPLE)
	DI	dependency injection --- setting the properties of a bean (by name, by type) by the container
	AOP	introduce new features between caller and callee without changing them --- add an aspect and it acts as a **proxy**
	Thymeleaf	Parallel to JSP for generating dynamic html (can be used with MVCController)
	Postman	REST CONSUMER --- it consumes data coming from REST API
	RestTemplate	API to write a java code similar to postman - REST CONSUMER

	REST	Web service that has mapped http methods . It returns DATA in json/xml format

Advantage	Inner class	inner class can access the private and all members of outer class
	Lambda	quickly implement functional interface INLINE where we want to pass an interface object
	Reflection	get Info about object --- methods,constructors, properties, super class, super interface -METADATA ---creating the object instance at **runtime** --- invoking methods at **runtime**
	Distributed n-tier	
	Jdbc	data can come to Java program so it can be easily processed
	Http	Web client web server communication Universally , RESTful Web Service
	Servlet	generate dynamic html , process data, DB connectivity , Session management
	JSP	it gets converted to servlet - so all servlet advantages PLUS easy to write UI code
	Hibernate	Java programmer is decoupled from DB --- talks to entities , that are **mapped** to TABLES
	JPA	Programmer is decoupled from different ORM APIs / programmer can work only with JPA even though underlying ORM changes
	Spring Framwework /Spring Boot	We get lot of functionality wrapped under simple APIs ---example JPA Repository --- automatically implemented --- MVC Controller (wrapping servlet controller) ---- REST controller (wrapping REST controller and container and **Providers)**

Implementation	inner class	implicit this of outer class is available , to create inner class object from outside of outer class new Outer().Inner() new Outer.Inner() Runnable obj = new Runnable (Public void run(){ …..})

	Lambda	Syntax Runnable obj = ()->{ impl of run } Java collections - **Stream** methods accept lot of functional interface objects foreach map filter sort
	Reflection	class Class , Method, Constructor, Field --- we cant create class Class ---JVM creates when class is loaded --- we can get it ------ obj.getClass() , Class.forName(classname) , clsobj = java.lang.String.class
	Distributed n-tier	
	JDBC	Statement PreparedStatement (pre compile , fast , ? Mark so easy for queries with variables, tield up with a query) Resultset = object holding rows (rs.next( )) Transaction = con.setAutoCommit(false) , con.commit() , con.rolloback() queries complete together or fail together (Account transfer example) SQLException = jdbc fails we get this exception (Checked exception) Driver --- type 4 Mysql connector -- translate from java to DB Jdbc url = url to connect to DB DML ---- API executeUpdate DQL -----API executeQuery DDL---- API execute
	Http	what http methods -, default , URL append , path variables, query parameters http status --- 200, 201, 400, 404, 405, 500 passing JSON in Response body Difference between GET and POST
	Servlet	HttpServlet , HttpSession ---(sessionID) cookies, urlrewriting (**Userwise Data** hitcount) --- invalidateSession() LOGOUT , LOGIN - we can preserve userwise data for all requests between login and login, discarded Serlvet lifecycle ----- init() , service() [doGet() , doPost()] , destroy ---

		automatically called by the container How many servlet objects are created by a container --- ONE shared by all requests RequestDispatcher = servlet chaining - request forward (include,forward)
	JSP	JSP ------------>Internal servlet (lifecycle _jspInit,_jspservice, _jspdestroy) scriplet <% %> --- _jspservice initialization expression <%! %> ------- directly in class output expression <%= %> --- part of out.write() Directives ----- page(for import , session,erropage), include(static page including), taglib (custom tags) Actions ---<jsp:include> <jsp:forward> <jsp:param > EL ---------$ shorthand to access request , session parameters SCOPES --- page , request, session, application } set attribute/ get attribute
	Tomcat deployment	webapps folder should have the WAR file web.xml Deployment Descriptor OR @WebServlet
	Hibernate /JPA	Entity (Mapping class), hibernate-cfg.xml (jdbc url, driver, uname,pass , hbm2DDLauto - CREATE,UPDATE , CREATE-DROP, mapping class names , connection pool) Life cycle of entity ---- transient , persistent, detached ,removed how the state transition happens---- @Id, @Entity, @Columns -primary key CRUD ---- save, saveorupdate, remove, find , query.list() Relationships ----- @onetoone , onetomany, manytoone , manytomany EXAMPLES
	Spring	@Configuration , @Controller , @RestController @Component, @Service, @Repository @Bean , @Entity @Lazy , @CrossOrigin, @Scope @Aspect @GetMApping , **@Autowired ---- Dependency Injection by name, by type** @RequestBody --- to pass json to java method @ModelAndView --- in MVC to set the view and pass model to request

		@SpringBootApplication --- it includes many annotations including @Component-scan To include beans from xml @ImportResources Application Context JPARepository findbyColumnName @Query (….) Difference between JPARepositary and CrudRepositary