Web Service

1. SOAP – Simple object Access Protocol
2. REST – Representational State Transfer (http methods)

Spring Boot is the easiest way to create RESTful web service

Terminologies related to web Services & Http Methods

1. End points
2. URI – Uniform Resource Information
3. Http Methods – to Do the CRUD Operation (Post/Get/Put/Delete http methods respectively)
4. Create/Insert, Read, Update, Delete – 95% web application

Terminologies Related to Spring Boot

1. @RestController
2. @SpringBootApplication
3. @RequestMapping – Generic Mapping which is used for any type of http request
4. @GetMapping
5. @PostMapping
6. @PutMapping
7. @DeleteMapping
8. @PathVariable
9. @RequestBody
10. Controller class
11. DAO Class /Repository (I) extend JPARepository or CrudRepository
12. Service Class
13. Entity Bean Class

Tool for API Testing

1. Using Postman (Stand-alone or web based)
2. Swagger (API Documentation) – CURL – Git bash – CLI /GUI

Different ways of creating spring boot project

1. Using spring initializr (start.spring.io) – maven/graddle based spring boot project (pom.xml/build.graddle)
2. Using STS (Spring Tool Suite is a Eclipse based IDE) – Spring Starter Project
3. Using SpringBoot CLI

SpringBoot is a opinionated framework – It’s based very popular java based framework called spring.

To configure a spring boot project, we can use a text based file called application.properties (key=value)

Spring – Framework of Frameworks (It supports other frameworks [struts, hibernate…] as well).

Spring-starter-web -- This is basic dependency to create RestFul web Service project.

**Hibernate – ORM Framework**

ORM – Object Relational Mapping (JAVA Object – RDBMS Entity - Mapping means connecting together)

Hibernate is a JPA implementation

JPA – Java Persistence API – IT’s actually a specification

JPA implementation

1. Hibernate ( A Very popular – Caching, connection pooling, load balancing, HQL )
2. EclipseLink
3. Generics
4. IBatis

Official URL – hibernate.org

JPA –

Serialization – It is the process storing state of an object to a permanent storage (Flat file, .txt,.rtf)

Public class Employee {

Private int id;

Private String name;

Private String email;

Constructors , getter & setters;

}

Main() {

Employee emp = new Employee(100, “abc”, [abc@gmail.com](mailto:abc@gmail.com));

}

Serializable – Is a Marker Interface

Marker Interface – An Interface with no abstract method

Oracle Java 8 SE API documentation url - <https://docs.oracle.com/javase/8/docs/api/>

Persistence (javax.persistence )

Persistance = It’s a process of storing the state of an Object to a RDBMS Entity (Table)

Static, transient (member properties marked with these keyword will not be serialized /persisted)

JPQL – Java Persistence Query Lang (DB Independent Query)

EntityManager – It’ll help you to manage the entities

1. In java, everything is object (except primitives)
2. In RDBMS, everything is entity (table, view, query, stored procedure, functions, sequence, triggers, schema)

In 2001, Gavin King introduced Hibernate – Session & SessionFactory Interfaces

JDBC – Java Database Connectivity API (java.sql package – Interfaces & Classes)

Two important configuration files –

1. Hibernate.cfg.xml (Hibernate configuration file – details about db connection)
2. <entity>.hbm.xml (Hibernate Mapping file) – This file is optional if annotations are used in entity bean class

XML is case as well as space sensitive. –

To connect with a database, we need following details

1. Db url : (<http://www.google.com>)

To Create a Maven based project

mvn archetype:generate -DgroupId=com.revature -DartifactId=hibernatedemo -DarchetypeArtifactId=maven-archetype-quickstart -DinteractiveMode=false

  <dependency>

            <groupId>org.hibernate</groupId>

            <artifactId>hibernate-core</artifactId>

            <version>5.3.7.Final</version>

        </dependency>

        <dependency>

            <groupId>com.h2database</groupId>

            <artifactId>h2</artifactId>

            <version>1.4.200</version>

        </dependency>

        <dependency>

            <groupId>junit</groupId>

            <artifactId>junit</artifactId>

            <version>4.12</version>

            <scope>test</scope>

        </dependency>

Hibernate.cfg.xml

<?xml version=*"1.0"* encoding=*"UTF-8"*?>

<!DOCTYPE hibernate-configuration PUBLIC

"-//Hibernate/Hibernate Configuration DTD 3.0//EN"

"http://hibernate.sourceforge.net/hibernate-configuration-3.0.dtd">

<hibernate-configuration>

<session-factory>

<property name="hibernate.connection.driver\_class">org.h2.Driver</property>

<property name="hibernate.connection.url">jdbc:h2:mem:test</property>

<property name="hibernate.connection.username">sa</property>

<property name="hibernate.connection.password"></property>

<property name="hibernate.dialect">org.hibernate.dialect.H2Dialect</property>

<property name="show\_sql">true</property>

<property name="hbm2ddl.auto">create-drop</property>

<mapping class="com.revature.myhibernate.dto.Employee"></mapping>

</session-factory>

</hibernate-configuration>

Entity Bean class – It’s a bean class which represents a database table

@Id – to specify the primary key column

@Entity – To specify the entity bean class

@Table –To specify name of the db table

@Column – To give name to the table columns

<dependency>

    <groupId>javax.xml.bind</groupId>

    <artifactId>jaxb-api</artifactId>

    <version>2.3.0</version>

</dependency>