**How to make hibernate project**

* Maken maven project(File🡪New🡪Maven project) 🡪 tick mark create a simple project (skip archetype selection) 🡪 click next 🡪 set Group Id(package name) , set Artifact Id (project name)🡪click on finish.
* Copy and Paste dependencies on pom.xml(for this go to maven repository) and then search hibernate core relocation 🡪 search for 5.6.15 version(copy dependency) AND search mysql connector java 🡪 search for 8.0.28 version (copy dependency) ….paste these dependency inside dependencies tag.
* In src/main/resource folder, create hibernate.cfg.xml file 🡪(right click on src/main/resource 🡪 new🡪other🡪search for xml file) select and name it hibernate.cfg.xml
* In src/main/java folder, create a package and inside that create a class(.java file where your main function will come).
* Make entity class(with .java extension) in src/main/java(inside the same package which you made earlier)
* Make hibernate mapping file(.hbm.xml extension) in src/main/resource.

**Framework**

What?

* A framework is a pre-written code which includes multiple libraries and tools that helps a programmer to build application faster because programmer does not have to make everything from scratch.
* A framework also set some rules and guideline for the programmer. For ex- how to organize your code into different files and folder and how common task(like database access using JPA , security using Spring security ) will be managed …so framework provides a structured way to build an application to a programmer.

Libraries – libraries provide specific functionality. For example 🡪 spring-boot-starter-web → Includes libraries for handling web requests (like Spring MVC, Jackson for JSON processing).

Tools – Embedded Tomcat server in Springboot.

(just for understanding)

Why?

* The aim of the framework is to help the programmer to avoid writing repetative code or to write the code from scratch everytime. So that programmer can focus more on building application rather than low-level stuff.

For ex- Java Collection Framework (If I want to store some data, one option is I make my own data-structure and then store and second option is I can use already built data-structure present in Java Collection Framework and store there. Second option helps the programmer to focus more on building application rather than implementing low-level stuff and choosing second option will be faster and efficient also.

Efficient why?

Because the code we find in framework will be written by team of experience engineer and optimised to its maximum capacity so framework code will be more optimised then the code we write.

Java framework are Hibernate, Spring.

**Types of framework**

1. Invasive Framework – which will force the programmer to extends or implements its classes or interface. For ex- Struts, EJB framework
2. Non-invasive Framework – which will not force the programmer to extends or implements its classes or interface. For ex- Spring, Hibernate

**POJO Class (plain old java object) –** POJO is a class which does not have any special restriction other than that imposed by the java.

Rules for making POJO class

1. class must be public – because hibernate and our class both will be in different package, so hibernate can access the class, it must be public.
2. It should not extend to any class or implements any interface
3. All data fields must be private.
4. Public Getter and setter for each data field.
5. There should be no argument constructor. (so that hibernate can make the object)

**Advantage of POJO is that we can develop light weight application.**

**Entity Class** –The class which is annotated with @Entity and which represents table in the database is called entity class.

Rules for making entity class

1. Class must be public (because our entity class and hibernate both are in different package, so hibernate can only access our Entity class if we use public access specifier.
2. Must have one field which represents primary key
3. Private data fields
4. Public getter and setter for each data field
5. Public No argument constructor ( hibernate uses no argument constructor to make object)

**ORM (Object relational Mapping)**

**What?**

* When we map our object into the table, this process is called Object-Relational mapping.
* ORM example- Hibernate

How?

* ORM tool provides developer a platform where we deal with the object instead of writing SQL queries. ORM tools we write sql queries for us and interact with the database.

**Specification Of ORM**

* Every entity class will represent a table in the database.
* Every data field of enity class will represent a column in the table
* Every Object of an entity class will represent a record in the table.

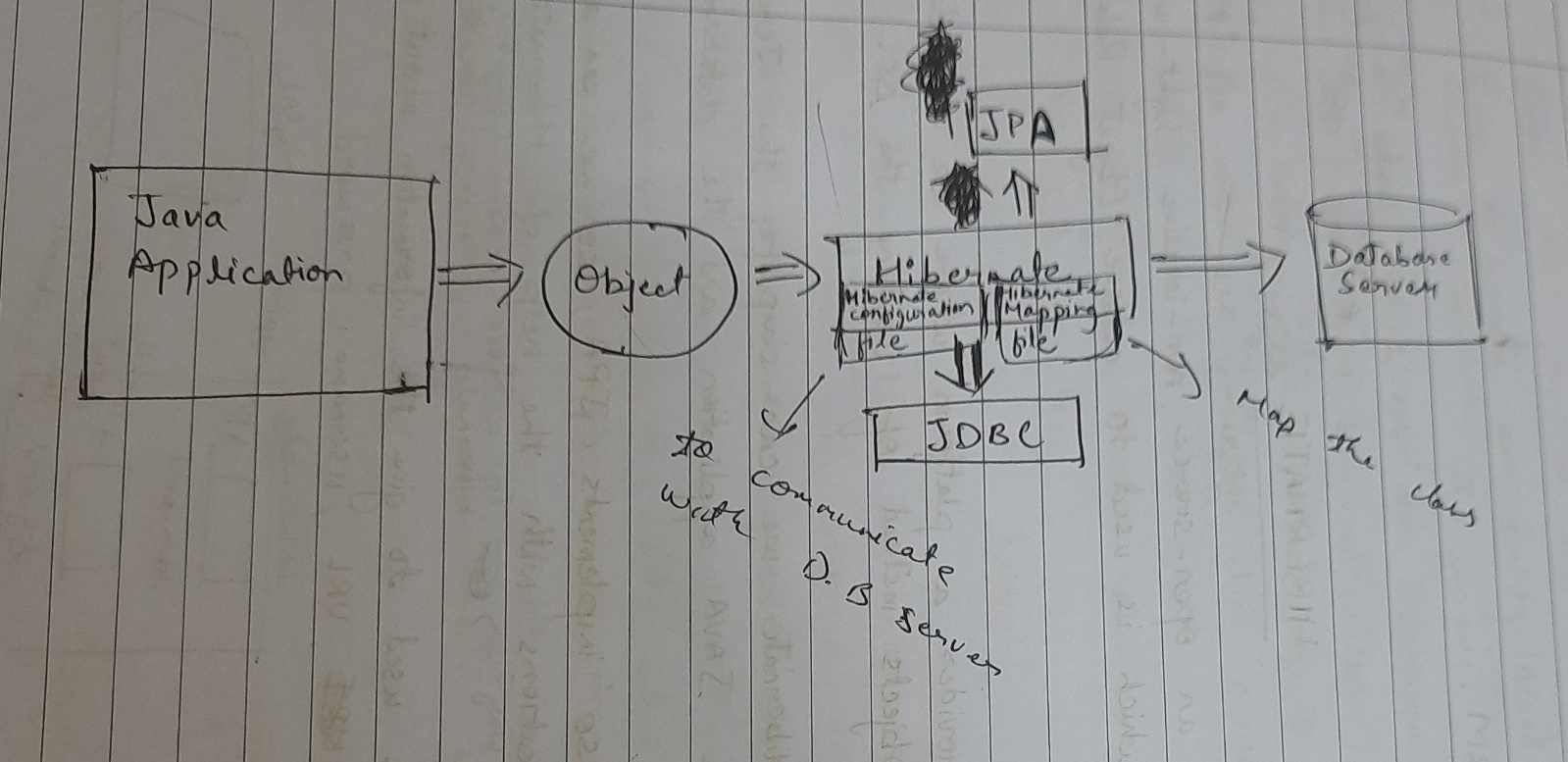
Q- What is **Hibernate**?

Hibernate is an open source, non invasive, light weight ORM tool which helps to convert an object into table.

**Advantages of Hibernate** (whatever the disadvantage of JDBC are, those are the advantage of hibernate)

1. NO boiler plate code
2. Automatic table creation
3. Automatic primary key generation
4. Cache mechanism(1st level cache and 2nd level cache) to reduce the traffic between java application and database server
5. If we have to fetch the data from multiple tables we do not have to write the complex join queries in hibernate.

**HIBERNATE ARCHITECTURE**



**Dialect**

* dialect is a class in hibernate.
* In dialect we tell the version of our database so that hibernate can generate version specific queries (as different databases have different SQL syntax)
* We need to tell dialect in the Hibernate configuration file.

**Hibernate Configuration File**

* In this file, we give all the details of our database to the hibernate, as hibernate is the one which will interact with the database.
* extension will be .cfg.xml and will be created in src/main/resources
* root tag will be <hibernate-configuration>
* child element of <hibernate-configuration> is <session-factory>
* child element of <session-factory> is <property> which is used to configure the properties like driver-class, url, username, password etc.



**HQL (HIBERNATE QUERY LANGUAGE)**

* HQL is database independent because

In HQL we use entity class name instead of table name.

* The reason to use HQL is ….sometimes we have to make query on the basis of name, age, gender hibernate do not provide any method for these fields, so at that time we have to write HQL query.

|  |  |
| --- | --- |
| SQL | HQL |
| select \* from employee | select e from employee e |
| select name from employee | select e.name from employee e |

org.hibernate.cfg.Configuration

Syntax- Configuration conf = new Configuration()

* Configuration is the class which is present in org.hibernate.cfg package
* It has two non-static method configure() and buildSessionFactory which helps us in loading the hibernate configuration file to hibernate and to build session factory from which we take out session, which helps in communicating with database

* configure() method is overloaded in two version

1. configure()

used when our file name hibernate.cfg.xml(default name)

return type of this method is configuration.

1. configure(String File)

used when our file is different then hibernate, pass whatever is our filename in String parameter. return type of this method is also configuration.

Syntax- conf.configure()

* buildSessionFactory() method

buildSessionFactory() method is a non-static factory or helper method which is used to to create implementation class object of SessionFactory interface.

Return type of buildSessionFactory() method is SessionFactory.

org.hibernate.SessionFactory

Syntax- SessionFactory fac = conf.buildSessionFactory() ;

* SessionFactory is a interface present in org.hibernate package.
* SessionFactory provides pool of session to us by which our java application will interact with the database.

**Difference between get() and load()**

|  |  |
| --- | --- |
| **get()** | **load()** |
| * get() is eager | * load() is lazy |
| * get() hit the database, if record is present it will return record otherwise null | * load() will return proxy object |
| * get() is slower as compared to load() (as it is hitting database and then returning record). | * load() is faster as compared to get() |
| * get() can be used to check the existence of a record. | * load() can be used to check the existence of a record |

**Hibernate Object States (Object lifecycle)**

Object goes through 4 States

1. Transient State –means object is not associated with session or hibernate is not managing the object. For ex- New object is created and initialized through constructor or setter method.
2. Persistent State- object is associated with active session and it is managed by hibernate, changes will be seen in database if commited.
3. Detached State – When we close the session, the object will be removed from session but record is present in database, this is called detached state.

If we change object’s value in detached State changes will not happen in database.

1. Removed State- When session’ s delete method is called the object is removed from database but present in session is called removed state.