ADVANCE -JAVA-THEORY

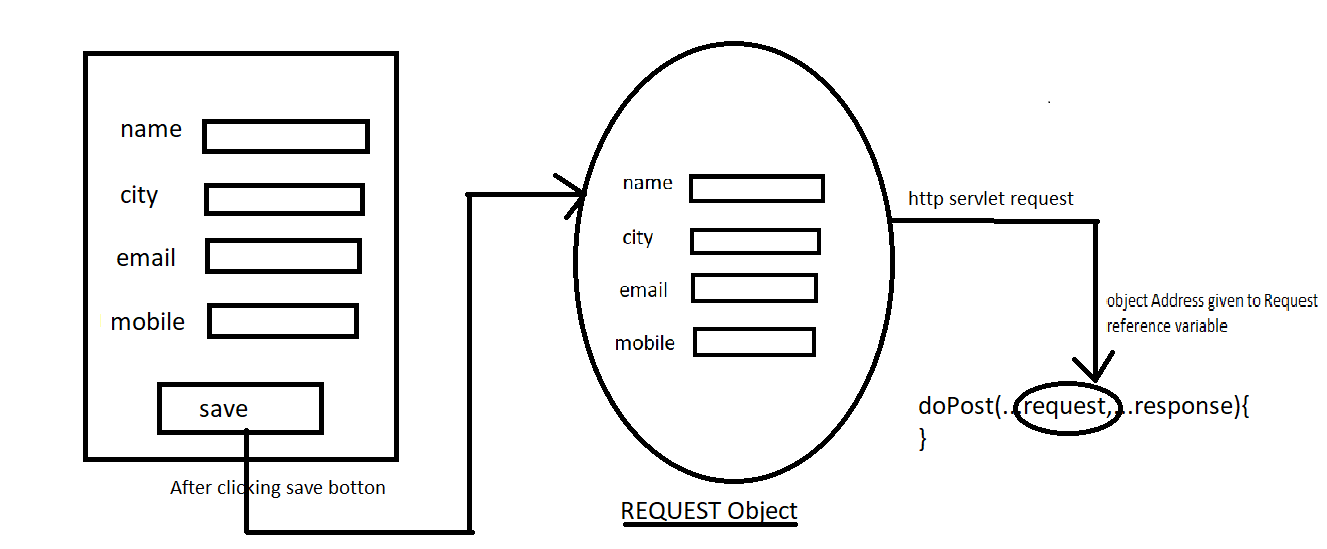
ADVANCE JAVA

1. Steps to create web application(without SQL & JDBC):

* Step 1 (installing tom cat):
  + New 🡪 others→ servers → Apache→ tomcat 9
  + Download tomcat → download zip→ unzip
  + Select tomcat on specified version in eclipse→ browse → tomcat → open until bin is found→ select → java jdk 1.8→ finish
  + Right click on server in bottom down section near console and start the server, or just click green play button.
* Step 2(dynamic web project):
  + Right click → dynamic web project instead of java project →name → web\_app\_1→ dynamic web module version (by default 4 if it wont work use 3.1).
  + Always create html files in src/main/webapp → new → other→ html→ file name→ (newRegistration)→ finish.
* Step 3(servlet):
  + Src/main/java → right click → others → servlets→ name→ (newRegistration)→ delete all comments.
  + Note: always use tomcat 9 in 2021 version as there are some problems with ver 10.
  + Note: @WebServlet("/newRegs") this should be same as that of the name which is present in form action.
  + Note: request is a reference variable which points to request object where all the data is stored as mentioned in the form.
  + Note: in html we use name attribute in input tag as it is similar to variables it stores variables and those can be used in backend.
  + Note: request.getParameters(“(city→ as it is written in name attribute in input tag inn html)”)
  + Here we use post method to send data to the sql so write all these code in post method in servelets.
  + Note: constructing login page using tables→ form→ table→ (tr→ td→td)\*4

1. Concept of POST/Get Request:

* Request is a reference variable where it points to request object where all the data is stored (i.e., the name attribute things with input present in html file).
* Serialization implementation is happening in the background.
* Concept:



1. Steps to create web application(without SQL & JDBC):

* Step 1 (installing tom cat):
  + New 🡪 others→ servers → Apache→ tomcat 9
  + Download tomcat → download zip→ unzip
  + Select tomcat on specified version in eclipse→ browse → tomcat → open until bin is found→ select → java jdk 1.8→ finish
  + Right click on server in bottom down section near console and start the server, or just click green play button.
* Step 2(dynamic web project):
  + Right click → dynamic web project instead of java project →name → web\_app\_1→ dynamic web module version (by default 4 if it wont work use 3.1).
  + Always create html files in src/main/webapp → new → other→ html→ file name→ (newRegistration)→ finish.
* Step 3(SQL Query):
  + Create sql database in MYSQL
  + Create database 4pm\_demo\_db\_1;
  + Use 4pm\_demo\_db\_1;
  + Create table registration(FirstName varchar(20), city varchar(20), email varchar(20), email varchar(20), mobile varchar(10));
  + Select \* from registration;
  + insert into registration values('Pankaj', 'Bangalore', 'pankaj@gmail.com', ‘9496415549’);
* Step 4 (Copying SQL connector):
  + Copy SQL connector file and paste in src/main/webapp/web INF/lib.
  + Note: always write 🡪 Class.forName(“com.mysql.jdbc.Driver”); before establishing connection.
* Step 5(servlet):
  + Src/main/java → right click → others → servlets→ name→ (newRegistration)→ delete all comments.
  + Here use same Connection 🡪 Statement 🡪 request.getParameters(“(\_)”)→ assign it to variable and use ‘”+var name+”’ to give into SQL Query inside stmt.updateQuery(“”);
  + Note: always use tomcat 9 in 2021 version as there are some problems with ver 10.
  + Note: @WebServlet("/newRegs") this should be same as that of the name which is present in form action.
  + Note: request is a reference variable which points to request object where all the data is stored as mentioned in the form.
  + Note: in html we use name attribute in input tag as it is similar to variables it stores variables and those can be used in backend.
  + Note: request.getParameters(“(city→ as it is written in name attribute in input tag inn html)”)
  + Here we use post method to send data to the sql so write all these code in post method in servelets.
  + Note: constructing login page using tables→ form→ table→ (tr→ td→td)\*4

1. Servlets:

* Servlets is a java class.
* It is a subclass of HTTP servlets; it is used to perform Back-end coding of the application.

1. Note: Whenever we run servlet it will always execute get method.
2. Note: whenever we want to retrieve data from database then we should always use updatequery method and 🡪assign to local reference variable 🡪 PrintWriter out = response.getWriter(); 🡪 out.println() to write an html code to display in the front end.
3. Note: during validation of sql data like login page we use SQL Query i.e.,🡪 select \* from login where email=’”+email+”’ and password=’”+password+”’.
4. Note: anything inside WEB-INF folder can only be run by servlet 🡪 request.getRequestDispachther(“location with /”) 🡪 rd.forward(request, response) – if we want to go to given page, if we want to same or backward page use rd.include(request, response);🡪if its run directly it will give 404 error.
5. Note: index.html → is the first page of html always , even if we don’t write index.html in web link also it will point to index.html as it is default start/home page.

PROGRAM-TOPICS:

* Core java SQL Injections:

1. Inserting data into SQL (normal)
2. Deleting data from SQL (normal)
3. Updating data from SQL(normal)
4. CRUD- all 4 operations using scanner class and using ‘”+variable name+”’ 🡪 Note: during retrieving of data use while loop with condition as result.next() and sysout(result.getString(1---n)).

* Advance Java Without SQL:

1. Normal HTML 🡪 servelet 🡪 String name =request.getParameter(“name”);🡪 print entered values into console.

* Advance Java with SQL:

1. Registration (SQL, HTML & Servlet)🡪 Class.forName(“com.mysql.cj.jdbc.Driver”);
2. update (SQL, HTML & Servlet)🡪 Class.forName(“com.mysql.cj.jdbc.Driver”);
3. delete (SQL, HTML & Servlet)🡪 Class.forName(“com.mysql.cj.jdbc.Driver”);
4. Retrieving Data and printing into html table 🡪 use updateQuery and assign it to local reference variable 🡪 PrintWriter out = response.getWriter(); 🡪 out.println() to write an html code to display in the front end.
5. Login page simple to print welcome If valid and invalid if its wrong into console🡪 during validation of sql data like login page we use SQL Query i.e.,🡪 select \* from login where email=’”+email+”’ and password=’”+password+”’.
6. Login page where Index (login page)🡪servlet (login Servlet)🡪index (welcome page), in if-condition write forward code i.e.,

RequestDispatcher rd=request.getRequestDispatcher("WEB-INF/views/welcome.html");

rd.forward(request, response);

and in else condition we write include code i.e.,

RequestDispatcher rd=request.getRequestDispatcher("index.html");

rd.include(request, response);