



# Advanced Java

*Trainer: Nilesh Ghule*



# Transactions

## MySQL

START TRANSACTION;

  dml1;

  dml2;

COMMIT;

or

ROLLBACK;

## JDBC

con.setAutoCommit(false);

execute multiple dml statements

e.g. Stmt1.executeUpdate()

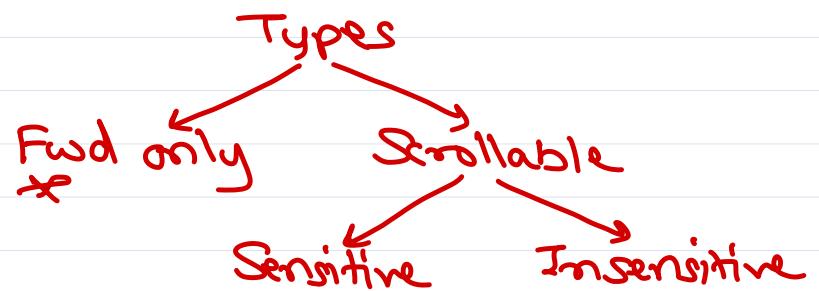
Stmt2.executeUpdate()

→ con.commit();

→ con.rollback();

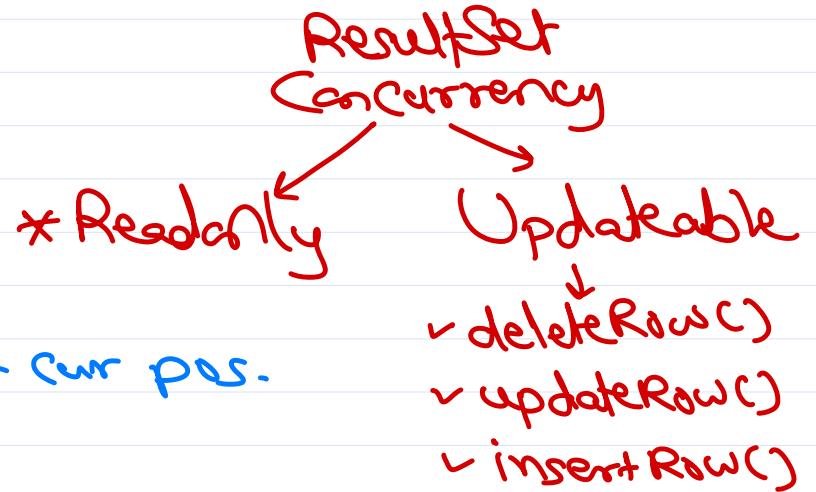
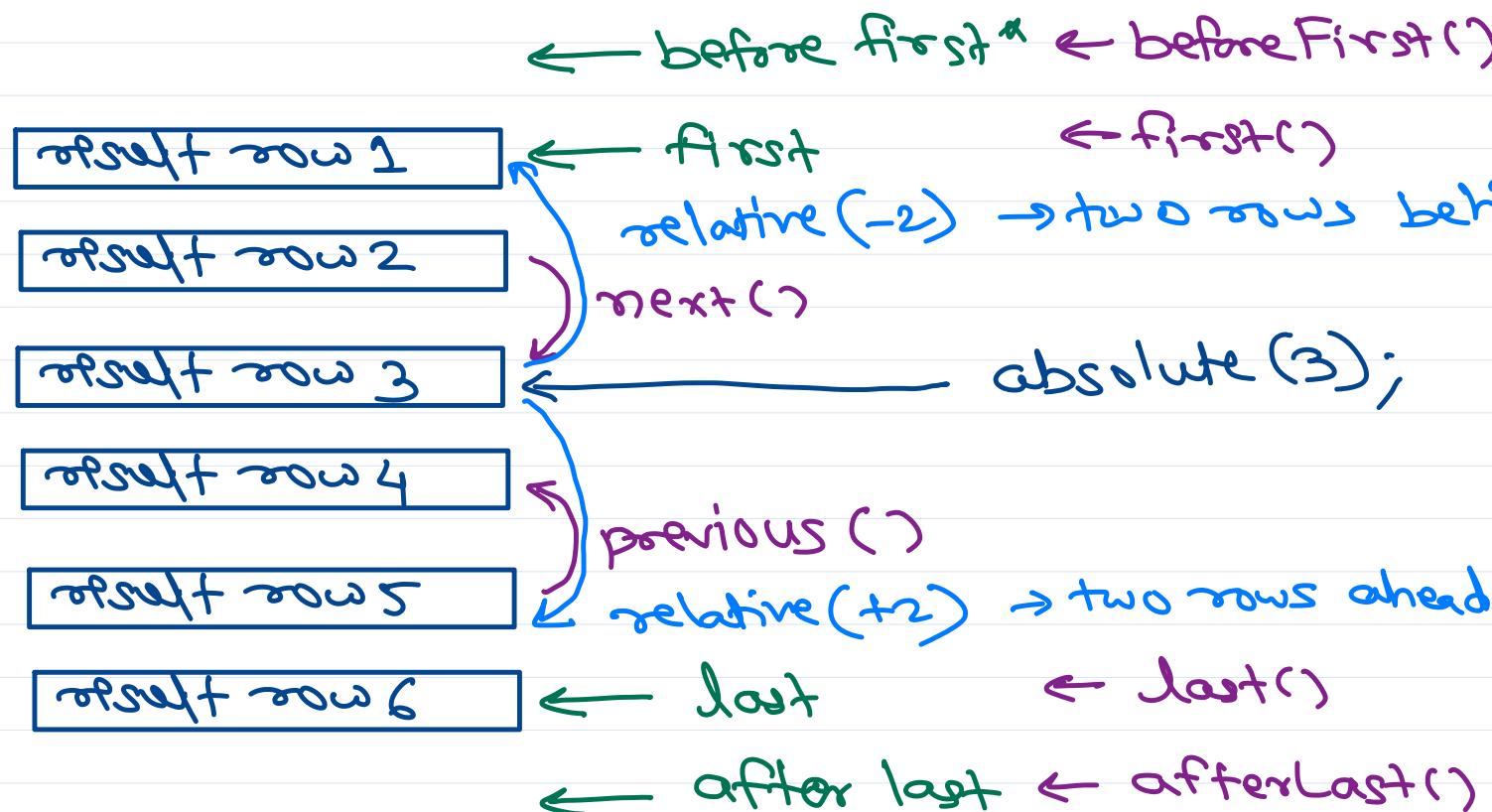
Note: Spring @Transactional internally calls above jdbc methods to manage the tx automatically.

# ResultSet



**sensitive** → while  $\text{rs}$  is in process, if any changes done in db, they will be immediately available to  $\text{rs}$ .

**insensitive** → need to close & create new  $\text{rs}$  to get the latest changes from db.



## HTTP protocol

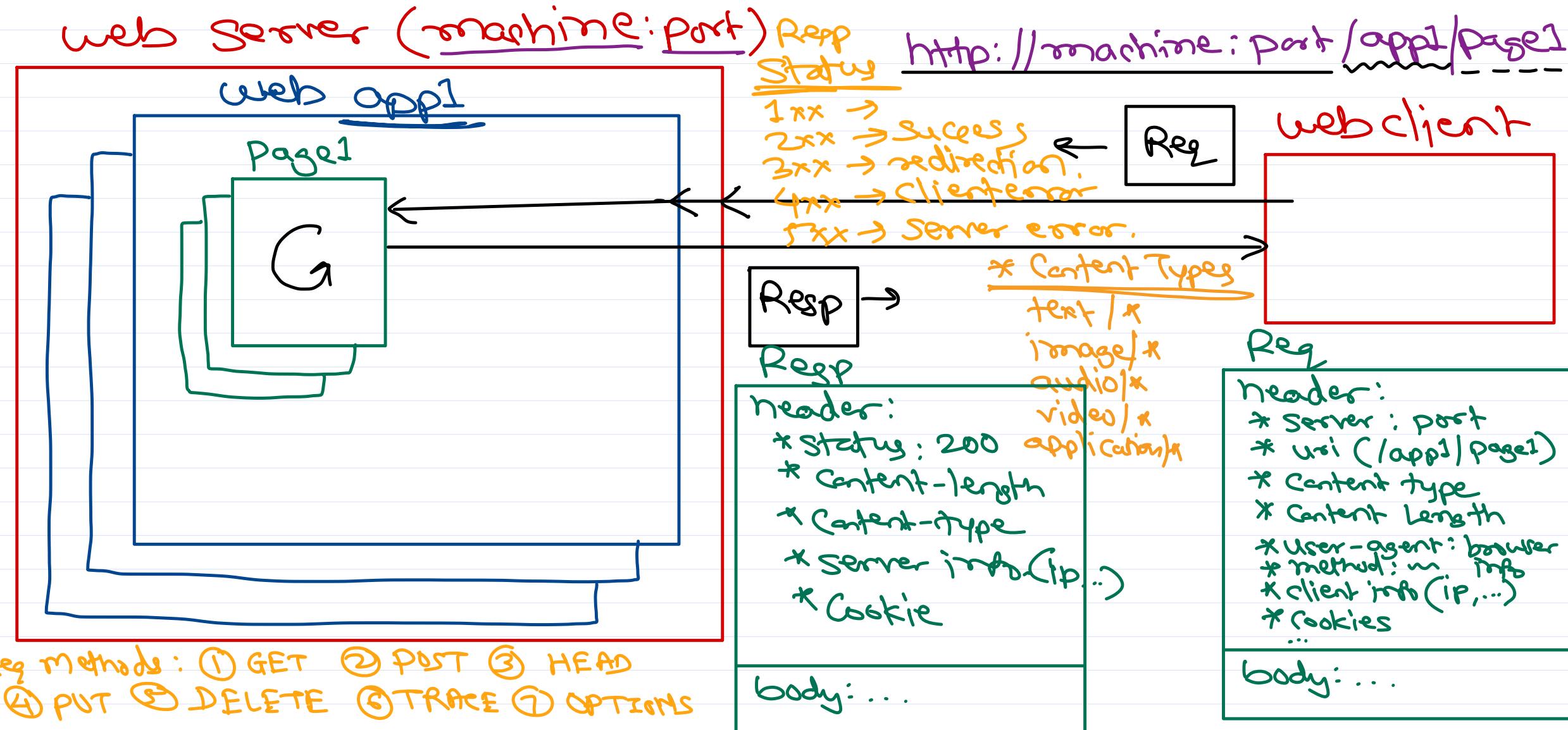
- ① Server → program that provides service(s).
- ② Client → program that consume service(s).

- ③ Web server → program that allows to run one/more web appn in it.  
e.g. Apache (PHP), tomcat (Java), IIS (.NET/ASP), JBoss (Java), ...  
\* Java web server → allows to run one/more Java web appn.
- ④ Web client → program that access web appn.  
e.g. browsers, mobile apps, ...
- ⑤ Web appn → made up of multiple web pages.  
Web pages → static page or dynamic page
  - ✓ static page → HTML pages (served to client as it is).
  - ✓ dynamic page → Executed on server & produces HTML/XML/JSON output that is sent to client.

- \* HTTP protocol → defines interaction b/w web client & web server.
- ① Request response model. → Req sent by client & then only resp generated & sent by server.
  - ② Connectionless protocol. → for each req new conn may be established by client & closed when resp is sent.
  - ③ stateless protocol. → by default, server do not remember any info/state of clients.



# HTTP protocol



# HTTP Protocol

## Request Methods

- ① GET : gets a page from server.
  - data is sent (if any) in url (no body).
- ② POST : posts data (in form) to server.
  - data is sent in body
- ③ HEAD : Like GET but get only resp header
  - to get info about page/resource.
- ④ PUT : Put a resource on server
- ⑤ DELETE : Delete a resource from Server.
- ⑥ TRACE : Used in debug issue
  - send req back as resp (loop back)
- ⑦ OPTIONS : Check which req methods are supported by resource/ page.

# Servlet

- \* Servlet is a Java class that is executed on server when req is received from client & produces resp that is sent to client.
- \* One of the way to implement dynamic pages in Java web appIn.

```
@WebServlet("/hello")
public class MyServlet extends HttpServlet {
    @Override
    public void doGet(HttpServletRequest req,
                      HttpServletResponse resp) throws ServletException, IOException {
        resp.setContentType("text/html");
        PrintWriter out = resp.getWriter();
        out.println("<html>...");
        out.println("<body>");
        out.println("<h1>Hello, Dmc! </h1>");
        out.println(new Date());
        out.println("</body>");
        out.println("</html>");
    }
}
```

Servlet is a specs given in form of interfaces & classes.



*Thank you!*

Nilesh Ghule <[nilesh@sunbeaminfo.com](mailto:nilesh@sunbeaminfo.com)>