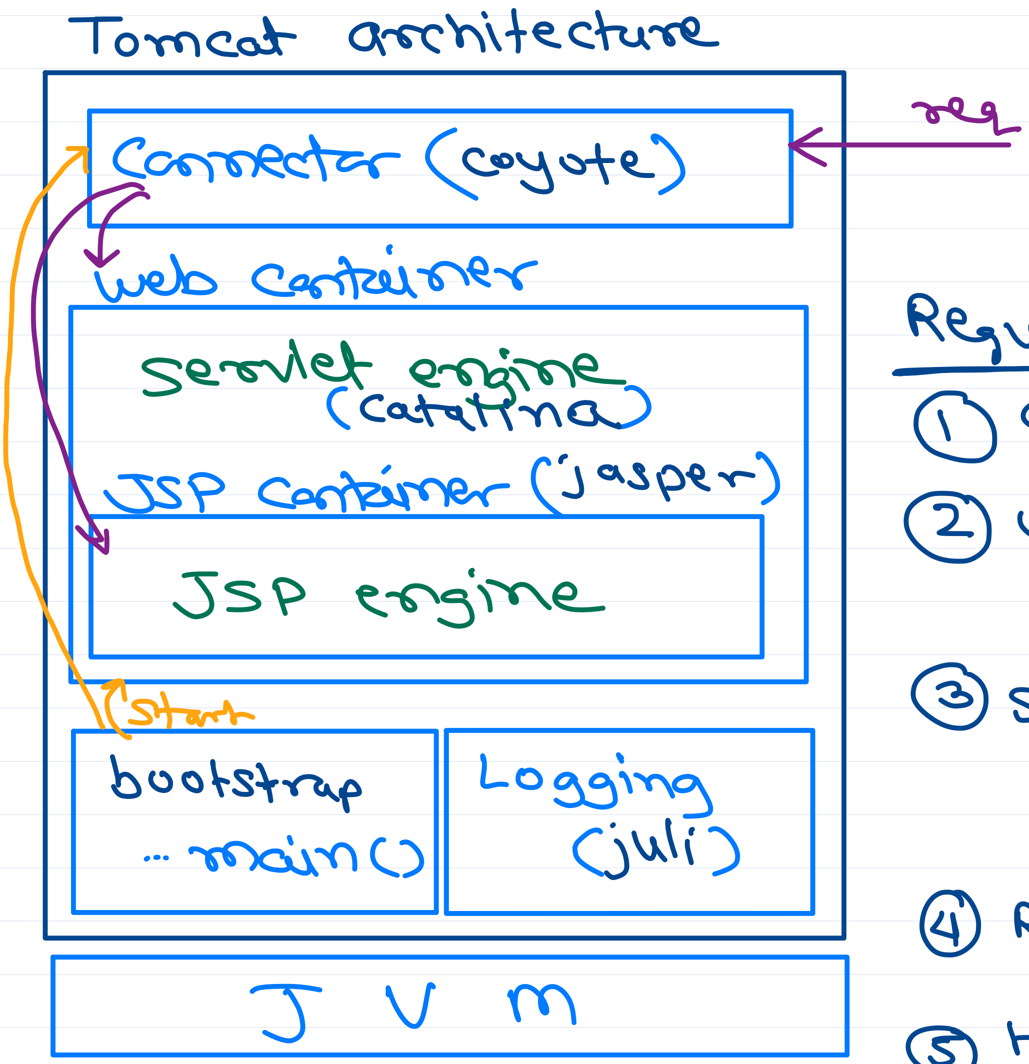
Java EE™

Advanced Java

Trainer: Nilesh Ghule

Tomcat architecture



Key parts:

1. Connector (accept reqs)
2. web container (req processing)
3. web applns running in tomcat.

Request Processing

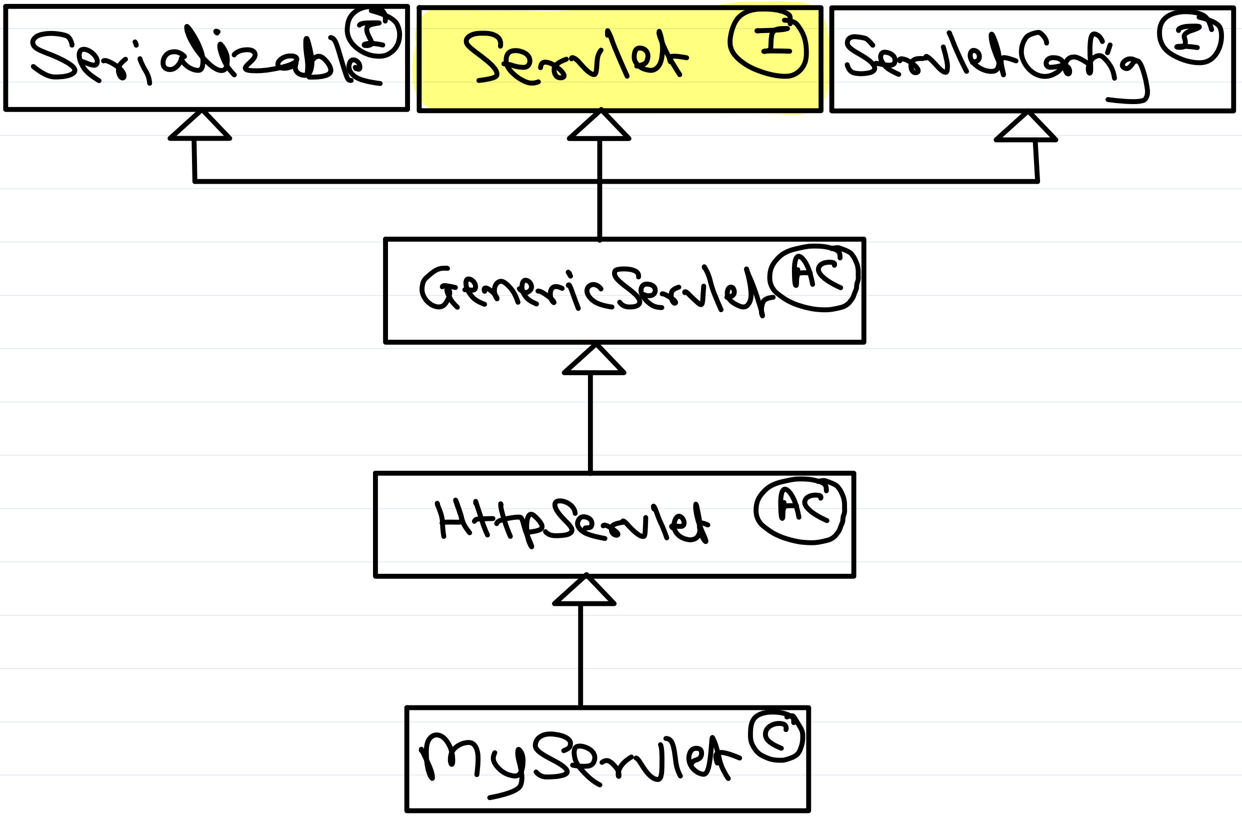
1. accept request
2. url mapping
   ↳ @WebServlet or web.xml
3. servlet object
   ↳ created on 1st req
   ↳ accessed from pool on next req
4. Response generation
   ↳ doXyz() method
5. HTTP response

Thread model

- ✓ tomcat creates a collection of threads during startup
  - executor thread pool.
- ✓ on each req to the connector, a thread from pool is assigned to process that request.
- ✓ upon completion, thread returns back to pool.
- ✓ max num of threads in pool can be configured in server.xml.

Java Servlets

Servlet is a java class that is executed in java web server when req is received from client and produces response that is sent back to the client.



Servlet interface: ← info/metadata of servlet

① init() → arg: ServletConfig
- when 1st req received for a servlet, obj is created and init() called by web container.
- programmer should override, if any one time initialization is to be done. e.g. JDBC connection, ...
- If failed, must throw ServletException. Now servlet reqs will not be processed further.

③ destroy()
- when appln is undeployed or web server shutdown, destroy is called by web container.
- programmer should override, if any one time de-initialization is to be done. e.g. close connection.

② service() →

Java Servlets

* Servlet interface:

② service()

- called for each req by web container.
- programmer should override it to process the req and produce the response.

* GenericServlet class

- provides default impl of init() & destroy().
- keeps service() abstract.

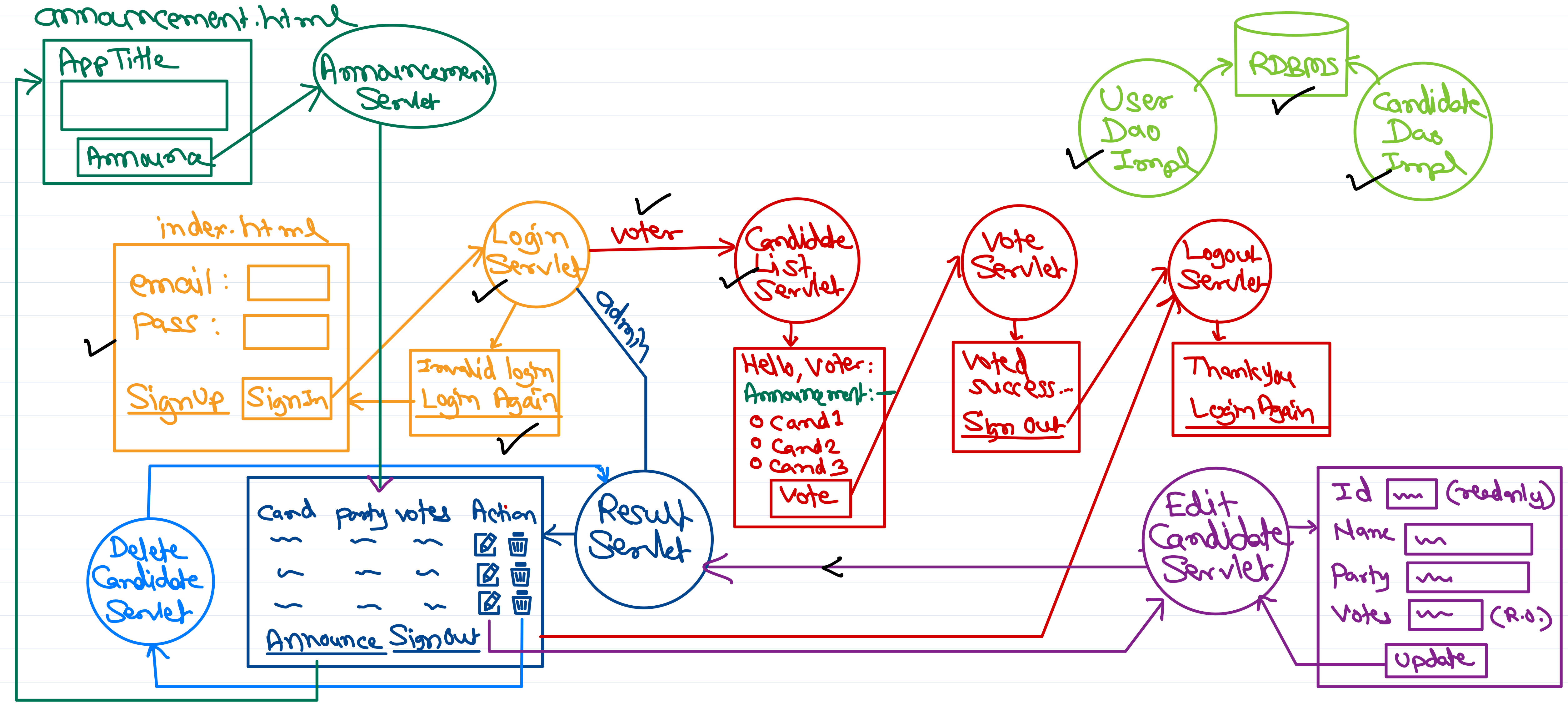
* Protocol independent servlet.

* HttpServlet class

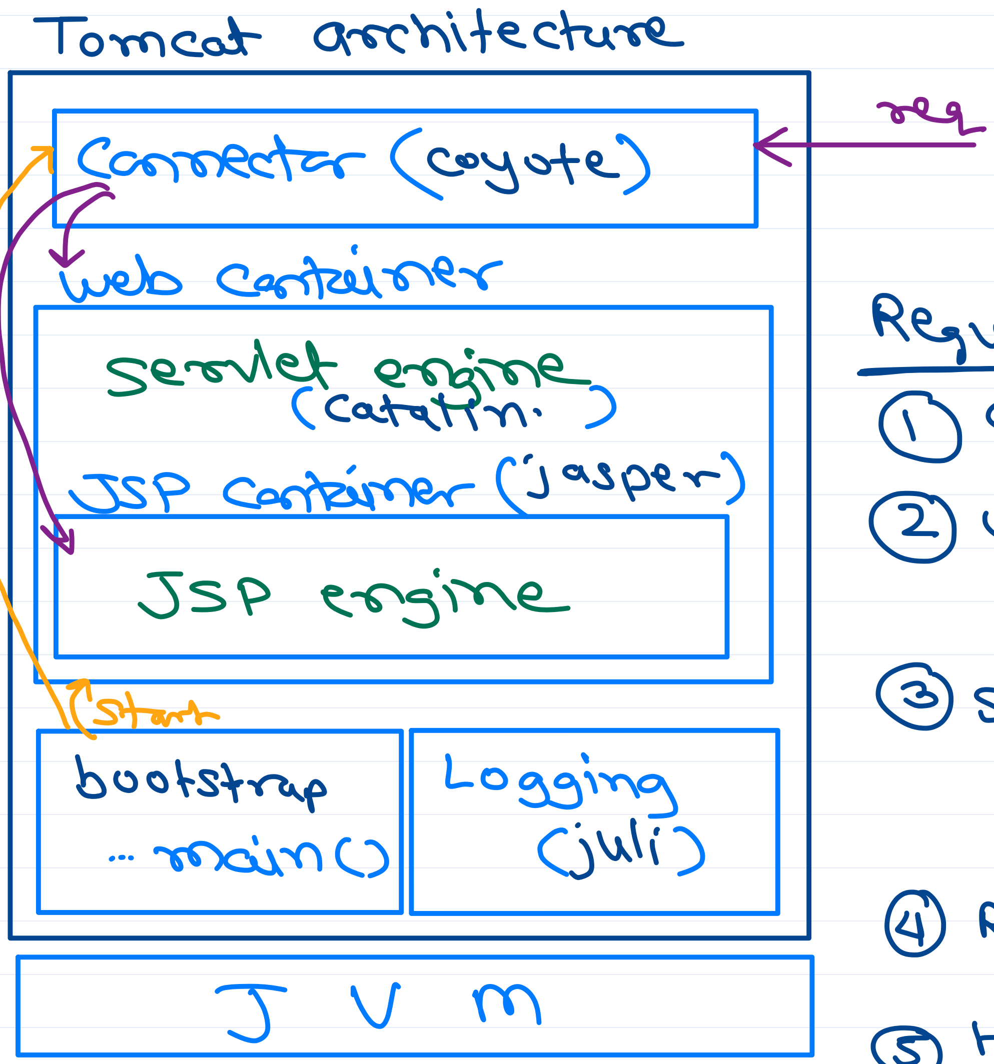
- class to handle HTTP requests.
- it overrides service() and internally calls doGet(), doPost(), doHead(), ... methods depend on current req. method.

```
// predefined class 2
class HttpServlet extends
                    GenericServlet {
  @Override
  void service(req, resp) ... {
     String m = req.getMethod();
     if(m == "get")
         doGet(req, resp);
     else if(m == "post")
         doPost(req, resp);
     else ...
  }
  void doGet(req, resp) { throw ... };
  void doPost(req, resp) { throw ... };
  ...
}
```

Election Management



Tomcat architecture



Key parts:

1. Connector (accept reqs)
2. web container (req processing)
3. web applns running in tomcat.

Request Processing

1. accept request
2. url mapping
   - @WebServlet or web.xml
3. servlet object
   - created on 1st req
   - accessed from pool on next req
4. Response generation
   - doXyz() method
5. HTTP response

Thread model

- tomcat creates a collection of threads during startup
  - executor thread pool.
- on each req to a connector, a thread from pool is asigned to process that request.
- upon completion, thread returns back to pool.
- max num of threads in pool can be configured in server.xml.

Request parameters

data from prev page controls or query string will come with req object.
It can be accessed in next servlet using.

```
String val = req.getParameter("param-name");

String[] vals = req.getParameterValues("param-name");
```

$\hookrightarrow$ checkbox, listbox,.

$\hookrightarrow$ textbox, radio, dropdown, *Thank you!*

Nilesh Ghule <nilesh@sunbeaminfo.com>