

Spring - Feb 22

Cookie:

Track you, to remember you

- remember me function

HTTPS

URL

- the advantage to use parameters:
 - GET request can not use body(rules)

<https://www.xyz.com/path/sdas/213?name=jack&ter=321>

protocol | host name | path | (query)parameters

server tries to find a servlet in java

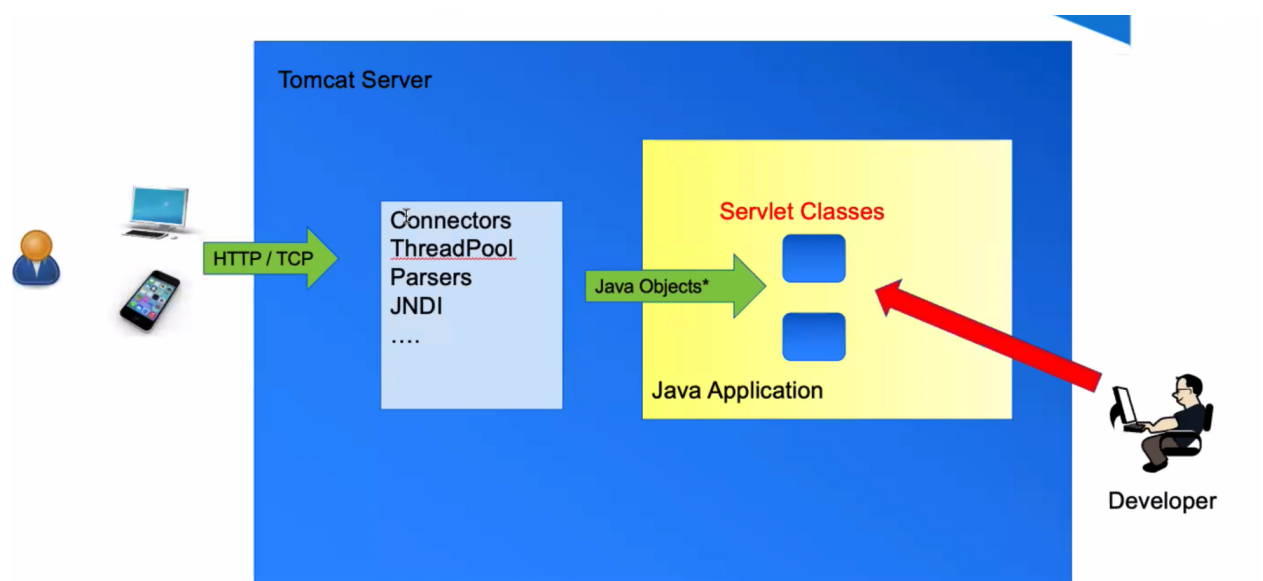
only path will be taken into consideration

the benefit of parameters in url → GET method does not allow to take with a body

? → question mark

& → divide parameter

Tomcat Server



Tomcat server will convert that HTTP requests into Java objects.

- Two objects: request and response
- we write Servlets to handle these two objects

Tomcat server deals with other things:

- how to connect with clients
- how to listen to ports

Servlet:

- how to handle request
- what kinds of contents of response to be sent

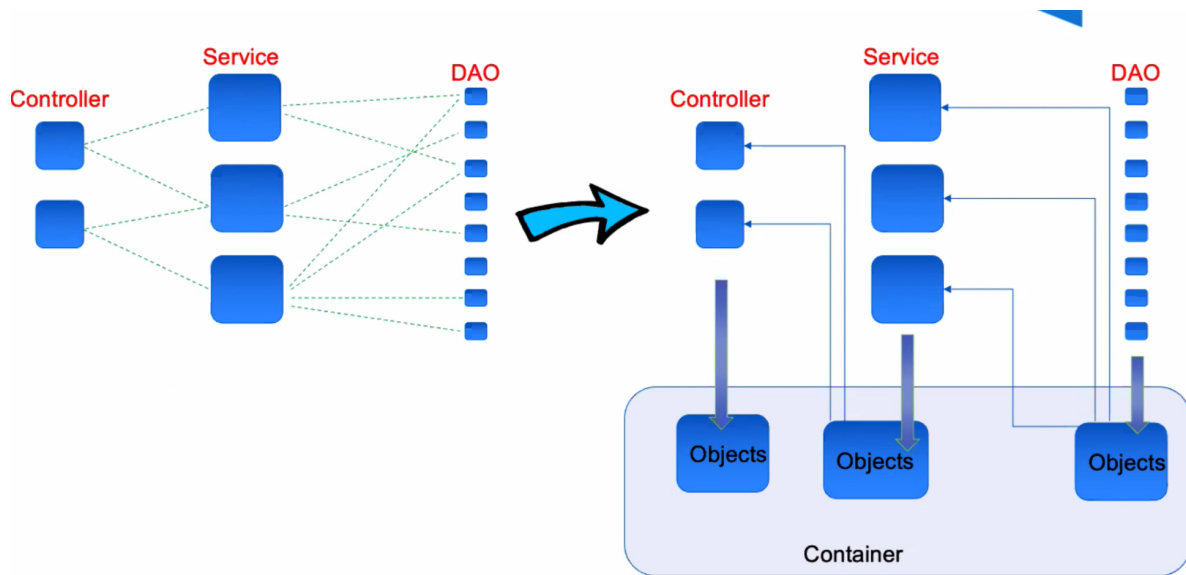
```
@WebServlet(name = "LoginServlet", urlPatterns = {"/login"})

public class LoginServlet extends HttpServlet {
    @Override
    protected void doGet(HttpServletRequest request, HttpServletResponse response) throws ServletException, IOException {
    }
}
```

Spring

IOC

- I need the class, just create to me



- The IoC container will create the objects, wire them together, configure them, and manage their complete life cycle from creation till destruction. These objects are called Spring Beans.
- The container gets its instructions on what objects to instantiate, configure, and assemble by reading the configuration metadata provided. The configuration metadata can be represented either by **XML**, **Java annotations**, or **Java code**

Material:

Tomcat Server

Apache Tomcat, which is often referred to as Tomcat Server, is an open-source Java Servlet Container developed by the Apache Software Foundation (ASF). Tomcat implements several Java EE specifications including Java Servlet, JavaServer Pages (JSP), Java EL, and WebSocket, and provides a "pure Java" HTTP web server environment in which Java code can run.

- **RPC(Remote Procedure Call)**: a function call to a remote server.
- **Java Servlet**: Java class to handle RPC on the server-side.
- Tomcat is an environment to run your web service, it provides low-level support such as making TCP connections, receiving requests from clients, finding the correct service to handle that request, and sending a response back.
- If you want to create a web service based on Tomcat Server, all you need to do is implement the logic to handle certain HTTP requests.

HTTP request and response

```
GET /doc/test.html HTTP/1.1
Host: www.test101.com
Accept: image/gif, image/jpeg, */*
Accept-Language: en-us
Accept-Encoding: gzip, deflate
User-Agent: Mozilla/4.0
Content-Length: 35

bookId=12345&author=Tan+Ah+Teck
```

Annotations for Request:

- Request Line: `GET /doc/test.html HTTP/1.1`
- Request Headers: `Host: www.test101.com`, `Accept: image/gif, image/jpeg, */*`, `Accept-Language: en-us`, `Accept-Encoding: gzip, deflate`, `User-Agent: Mozilla/4.0`, `Content-Length: 35`
- A blank line separates header & body
- Request Message Body: `bookId=12345&author=Tan+Ah+Teck`

```
HTTP/1.1 200 OK
Date: Sun, 08 Feb xxxx 01:11:12 GMT
Server: Apache/1.3.29 (Win32)
Last-Modified: Sat, 07 Feb xxxx
ETag: "0-23-4024c3a5"
Accept-Ranges: bytes
Content-Length: 35
Connection: close
Content-Type: text/html

<h1>My Home page</h1>
```

Annotations for Response:

- Status Line: `HTTP/1.1 200 OK`
- Response Headers: `Date: Sun, 08 Feb xxxx 01:11:12 GMT`, `Server: Apache/1.3.29 (Win32)`, `Last-Modified: Sat, 07 Feb xxxx`, `ETag: "0-23-4024c3a5"`, `Accept-Ranges: bytes`, `Content-Length: 35`, `Connection: close`, `Content-Type: text/html`
- A blank line separates header & body
- Response Message Body: `<h1>My Home page</h1>`