Persistent

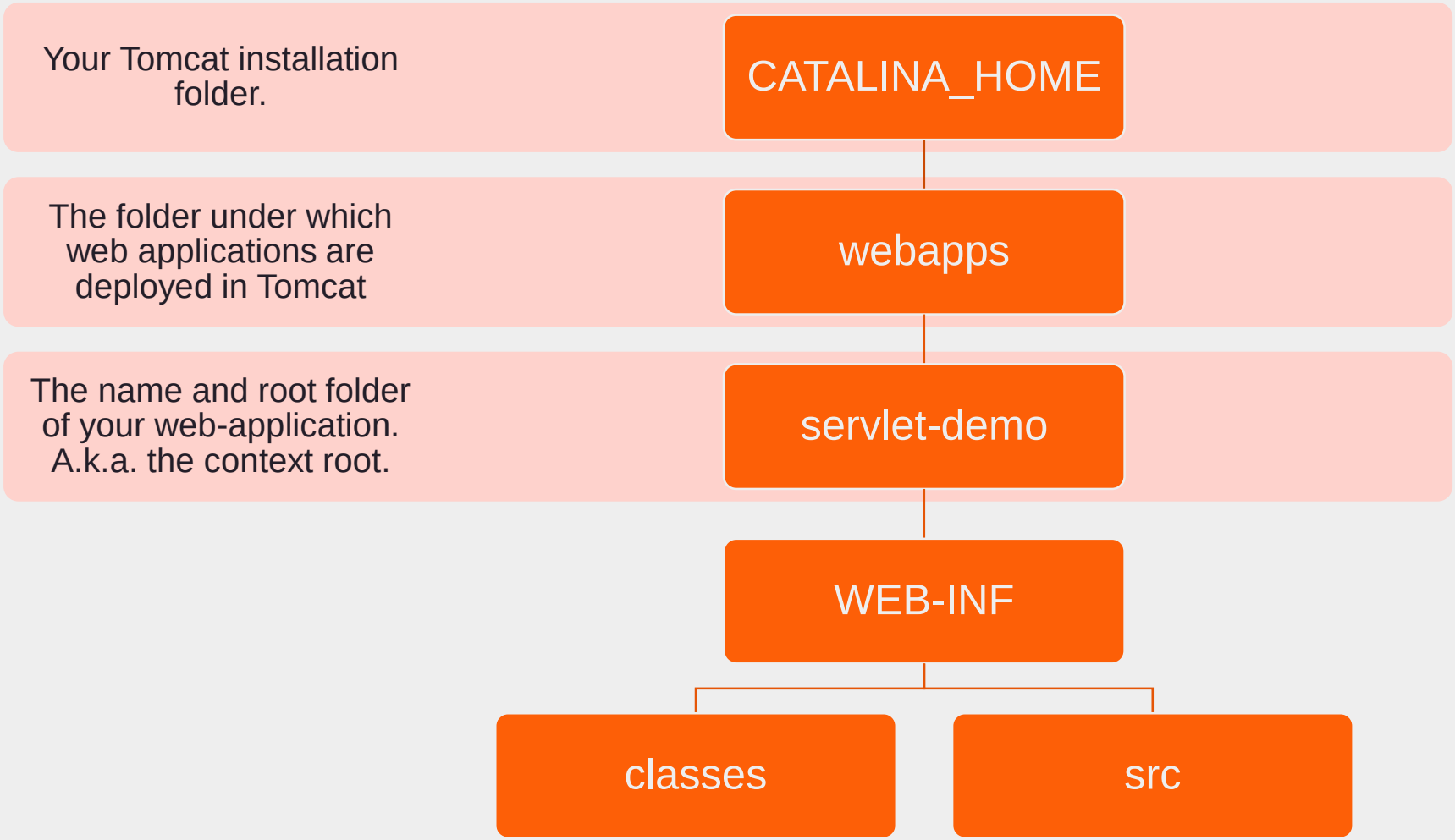
Servlet 2.5

Creating and deploying your first Servlet

Agenda

- Step by step guide to writing your first Servlet
- Configuration
- Deploying

Step 1 : the directory hierarchy



Step 2 : the servlet class

```
package com.demo;

import javax.servlet.*;
import javax.servlet.http.*;
import java.io.*;

public class HelloWorld extends HttpServlet {
    public void doGet(HttpServletRequest request,
    HttpServletResponse response)  throws IOException {
                response.setContentType(“text/html”);
                response.setStatus(200);
                PrintWriter out = response.getWriter();
                out.println("<html>");
                out.println("\t<body>");
                out.println("\t\t<p>" + new java.util.Date() + "</p>");
                out.println("\t</body>");
                out.println("</html>");
    }
}
```

Step 3 : the deployment descriptor

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<web-app xmlns=http://java.sun.com/xml/ns/j2ee
  xmlns:xsi=http://www.w3.org/2001/XMLSchema-instance
  xsi:schemaLocation="http://java.sun.com/xml/ns/j2ee
  http://java.sun.com/xml/ns/j2ee/web-app_2_4.xsd"    version="2.4">

    <servlet>
          <servlet-name>Demo Servlet</servlet-name>
          <servlet-class>com.demo.HelloWorld</servlet-class>
    </servlet>
    <servlet-mapping>
           <servlet-name>Demo Servlet</servlet-name>
           <url-pattern>/hello</url-pattern>
    </servlet-mapping>
    </web-app>
```

Step 4 : compile & access

javac -cp "<path to your Tomcat installation>\lib\servlet-api.jar"

-d WEB-INF\classes

WEB-INF\src\com\demo\ HelloWorld.java

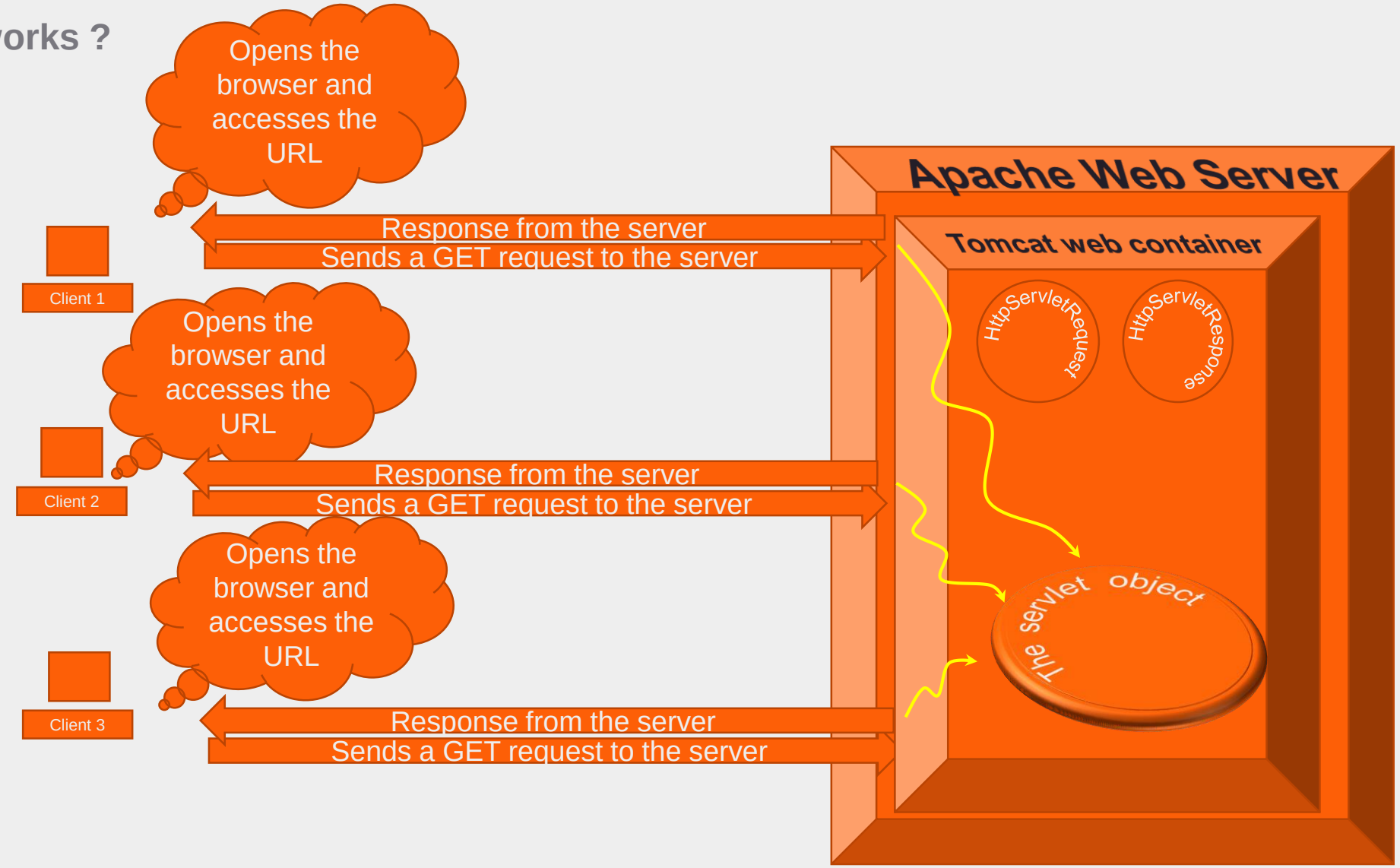
Start/restart the Tomcat server

Access the URL

http://<host-name or ip>:8080/servlet-demo/hello

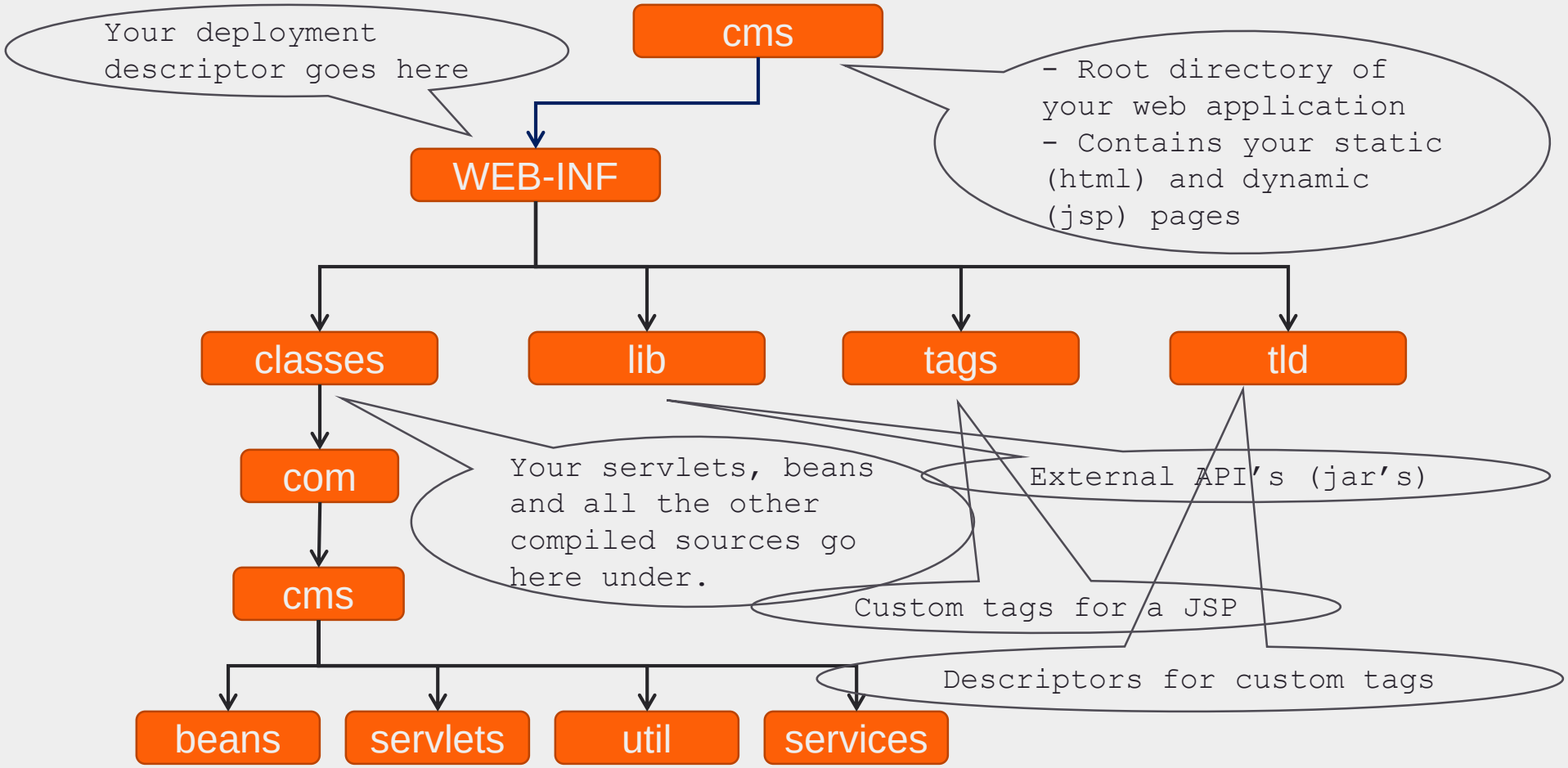
through your browser.

How stuff works ?



Persistent

The directory structure



Persistent

The servlet class

- Your servlet class extends from javax.servlet.http.HttpServlet
- It overrides the doGet method since the servlet will be accessed through a http GET.
- There are other do* methods in the HttpServlet class each of which corresponds to equivalent HTTP method.
- Instantiating the servlet and invoking the appropriate method is taken care of by the container.
- All of these methods have exactly two parameters viz. javax.servlet.http.HttpServletRequest and javax.servlet.http.HttpServletResponse
- HttpServletRequest and HttpServletResponse are server-side equivalents of a Http request and Http response.

The DD

- The *web.xml* contains a variety of configuration for your web application.

- In case of a servlet the programmer must use a unique name for the servlet to map it to its class and URL.

The deployment

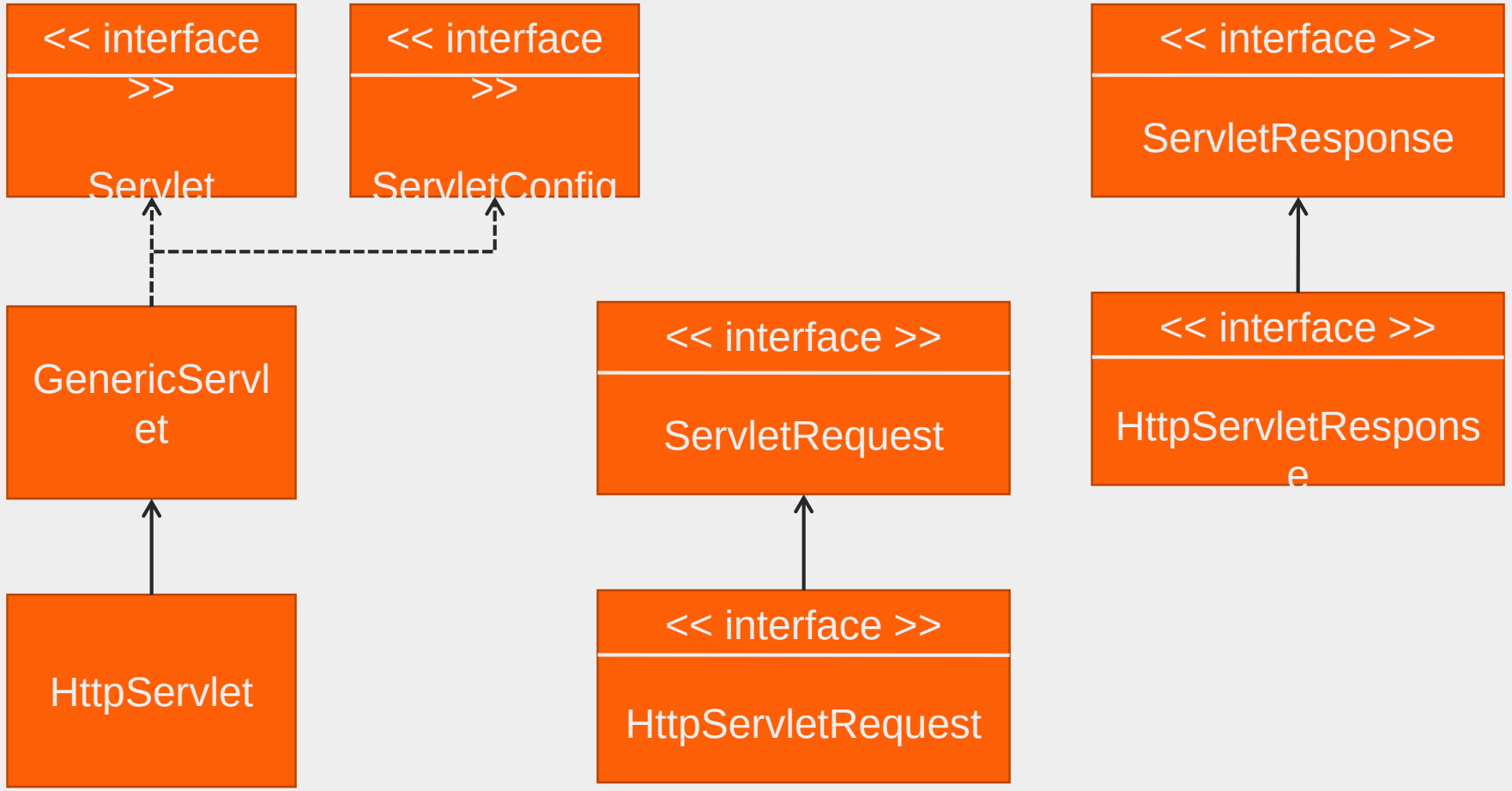
- As discussed earlier, Java EE requires a web-application’s compiled sources, pages and the deployment descriptor to be deployed under a standardized directory hierarchy.

- A web-application can also packed and deployed as a *war* (web application archive)

Accessing

- The servlet can be accessed through a web browser by appending the servlet‘s url as configured in the *web.xml* to the web-application’s context-root.
- The container is responsible for mapping the url request that it receives to the servlet’s class through its name.
- If the servlet has not already been initialized, then it is loaded, instantiated and initialized.
- The container starts a new thread which handles servicing of the request.
- Lastly, the container sends the servlet’s response back to the client.

Key API's



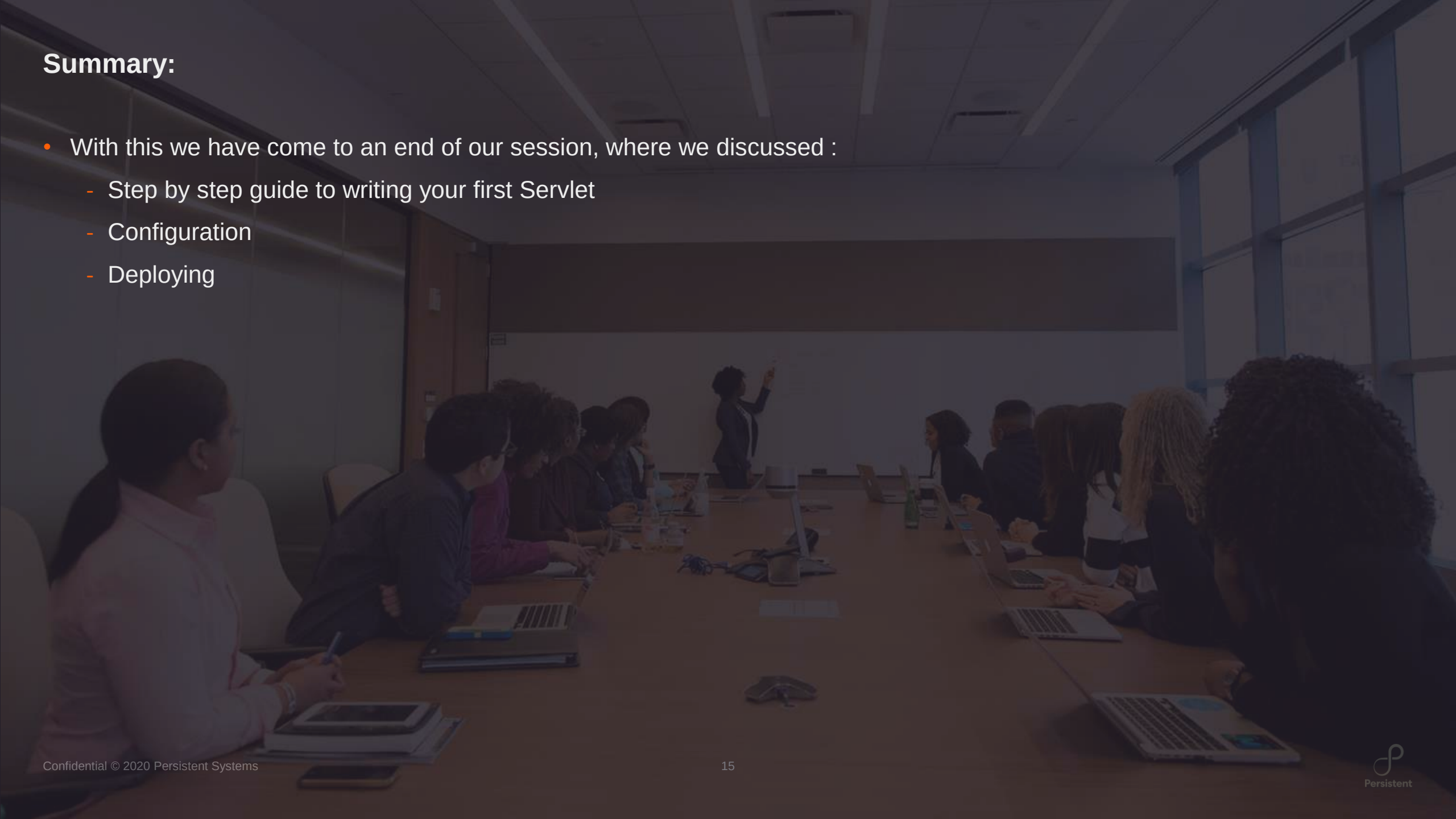
Persistent

Hands-on assignment

- Modify the servlet-demo web application to c0ntain a HTML form (index.html) with a text field and submit button. When users fill and submit the form the request should be sent to the HelloWorld servlet which will greet the user with a welcome message.

Summary:

- With this we have come to an end of our session, where we discussed :
  - Step by step guide to writing your first Servlet
  - Configuration
  - Deploying



Appendix

Thank You

Persistent

Thank you