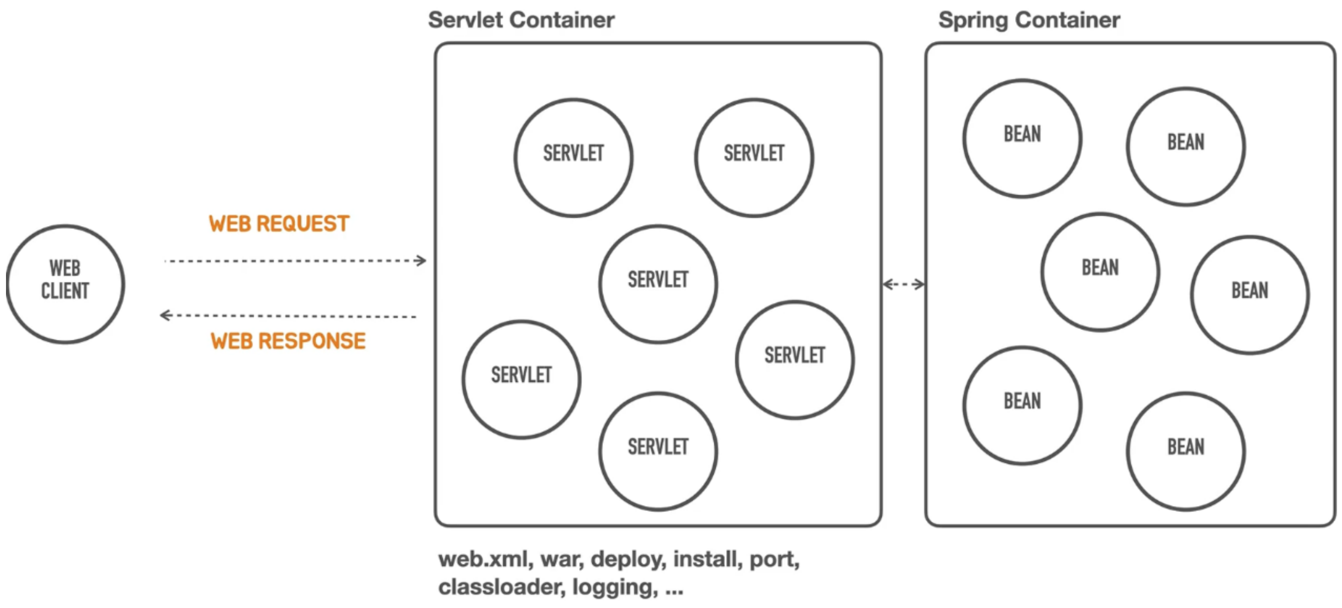
# 03. 독립 실행형 서블릿 애플리케이션

## Containerless 개발 준비



servlet container와 관련된 번거롭고 복잡한 작업들, 필요한 지식을 개발자가 신경쓰지 않고 spring container에 올라가는 component인 bean을 만드는거에만 집중해서 애플리케이션 개발하도록 해주기 위함

web.xml, servletcontainer 설치 등 필요없이 main method를 실행하는 standalone 방식으로 spring application을 실행시키고 싶은 것

```
package tobyspring.helloboot;

import org.springframework.boot.SpringApplication;
import org.springframework.boot.autoconfigure.SpringBootApplication;

@SpringBootApplication
public class HellobootApplication {

        public static void main(String[] args) {
                SpringApplication.run(HellobootApplication.class, args);
        }

}
```

@SpringBootApplication과 SpringApplication.run() 두개 뿐인데 spring app이 작동하고(servlet container 등), spring container도 뜸

## servlet container 띄우기

servlet container와 관련된 설치 배포 관리 등 작업을 신경쓰지 않도록 구성하는 작업을 해보자

STEP1
servlet container를 직접 설치하지 않고 어떻게 동작하게 만들것이냐, 직접 신경쓰지 않도록 할 것이냐?

servlet container를 설치하는 대신 standalone을 만들건데

standalone에서 servlet container를 알아서 띄워주게 하는 작업을 해야함

일단 servlet 하나를 만드는 작업을 해보자

servlet이란 java의 표준 기술

```
package tobyspring.helloboot;

import org.apache.catalina.startup.Tomcat;
import org.springframework.boot.SpringApplication;
import org.springframework.boot.autoconfigure.SpringBootApplication;
import org.springframework.boot.web.embedded.tomcat.TomcatServletWebServerFactory;
import org.springframework.boot.web.server.WebServer;
import org.springframework.boot.web.servlet.server.ServletWebServerFactory;


public class HellobootApplication {

        public static void main(String[] args) {

                //Tomcat servlet webserver
                //ServletWebServerFactory serverFactory = new TomcatServletWebServerFactory(); ->
                TomcatServletWebServerFactory serverFactory = new TomcatServletWebServerFactory();

                //servlet container
                WebServer webServer = serverFactory.getWebServer();

                // Tomcat servlet container
                webServer.start();
        }

}
```

```
 http -v :8080
GET / HTTP/1.1
Accept: */*
Accept-Encoding: gzip, deflate
Connection: keep-alive
Host: localhost:8080
User-Agent: HTTPie/3.2.1



HTTP/1.1 404
Connection: keep-alive
Content-Language: en
Content-Length: 682
Content-Type: text/html;charset=utf-8
Date: Tue, 11 Apr 2023 06:54:45 GMT
Keep-Alive: timeout=60

<!doctype html><html lang="en"><head><title>HTTP Status 404 – Not Found</title><style type="text/css">body
{font-family:Tahoma,Arial,sans-serif;} h1, h2, h3, b {color:white;background-color:#525D76;} h1 {font-size:
22px;} h2 {font-size:16px;} h3 {font-size:14px;} p {font-size:12px;} a {color:black;} .line {height:1px;
background-color:#525D76;border:none;}</style></head><body><h1>HTTP Status 404 – Not Found</h1><hr class="line"
/><p><b>Type</b> Status Report</p><p><b>Description</b> The origin server did not find a current representation
for the target resource or is not willing to disclose that one exists.</p><hr class="line" /><h3>Apache Tomcat
/9.0.73</h3></body></html>
```

## 서블릿 등록

서블릿 컨테이너에 웹 컴포넌트(서블릿)를 넣어보자

서블릿 컨테이너가 웹 클라이언트로부터 요청을 받으면 어떤 서블릿이 요청을 처리할지 결정하게 함 == mapping

간단한 자바코드로 서블릿 컨테이너를 띄우고 서블릿을 등록해서 동작시키는 실습

```
package tobyspring.helloboot;

import org.apache.catalina.startup.Tomcat;
import org.springframework.boot.SpringApplication;
import org.springframework.boot.autoconfigure.SpringBootApplication;
import org.springframework.boot.web.embedded.tomcat.TomcatServletWebServerFactory;
import org.springframework.boot.web.server.WebServer;
import org.springframework.boot.web.servlet.ServletContextInitializer;
import org.springframework.boot.web.servlet.server.ServletWebServerFactory;

import javax.servlet.ServletContext;
import javax.servlet.ServletException;
import javax.servlet.http.HttpServlet;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;
import java.io.IOException;


public class HellobootApplication {

        public static void main(String[] args) {

                //Tomcat servlet webserver
                //ServletWebServerFactory serverFactory = new TomcatServletWebServerFactory(); ->
                TomcatServletWebServerFactory serverFactory = new TomcatServletWebServerFactory();

                //servlet container
                //servlet container      object
                WebServer webServer = serverFactory.getWebServer(servletContext -> {
                        servletContext.addServlet("hello", new HttpServlet() {
                                @Override
                                protected void service(HttpServletRequest req, HttpServletResponse resp) throws
ServletException, IOException {
                                        //    ,
                                        resp.setStatus(200);
                                        resp.setHeader("Content-Type", "text/plain");
                                        resp.getWriter().println("Hello Servlet"); //getWriter.print: object to
string .
                                }
                        }).addMapping("/hello"); //mapping
                });

                // Tomcat servlet container
                webServer.start();
        }

}
```

```
 http -v :8080/hello
GET /hello HTTP/1.1
Accept: */*
Accept-Encoding: gzip, deflate
Connection: keep-alive
Host: localhost:8080
User-Agent: HTTPie/3.2.1



HTTP/1.1 200
Connection: keep-alive
Content-Length: 14
Content-Type: text/plain;charset=ISO-8859-1
Date: Tue, 11 Apr 2023 07:43:13 GMT
Keep-Alive: timeout=60

Hello Servlet
```

## 서블릿 요청 처리

```
package tobyspring.helloboot;

import org.apache.catalina.startup.Tomcat;
import org.springframework.boot.SpringApplication;
import org.springframework.boot.autoconfigure.SpringBootApplication;
import org.springframework.boot.web.embedded.tomcat.TomcatServletWebServerFactory;
import org.springframework.boot.web.server.WebServer;
import org.springframework.boot.web.servlet.ServletContextInitializer;
import org.springframework.boot.web.servlet.server.ServletWebServerFactory;
import org.springframework.http.HttpHeaders;
import org.springframework.http.HttpStatus;
import org.springframework.http.MediaType;

import javax.servlet.ServletContext;
import javax.servlet.ServletException;
import javax.servlet.http.HttpServlet;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;
import java.io.IOException;



public class HellobootApplication {

        public static void main(String[] args) {

                //Tomcat servlet webserver
                //ServletWebServerFactory serverFactory = new TomcatServletWebServerFactory(); ->
                TomcatServletWebServerFactory serverFactory = new TomcatServletWebServerFactory();

                //servlet container
                //servlet container      object
                WebServer webServer = serverFactory.getWebServer(servletContext -> {
                        servletContext.addServlet("hello", new HttpServlet() {
                                @Override
                                protected void service(HttpServletRequest req, HttpServletResponse resp) throws
ServletException, IOException {
                                        //    ,

                                        String name = req.getParameter("name");

                                        resp.setStatus(HttpStatus.OK.value());
                                        resp.setHeader(HttpHeaders.CONTENT_TYPE, MediaType.TEXT_PLAIN_VALUE);
                                        resp.getWriter().println("Hello " + name); //getWriter.print: object to
string .
                                }
                        }).addMapping("/hello"); //mapping
                });

                // Tomcat servlet container
                webServer.start();
        }

}
```

```
 http -v ":8080/hello?name=Spring"
GET /hello?name=Spring HTTP/1.1
Accept: */*
Accept-Encoding: gzip, deflate
Connection: keep-alive
Host: localhost:8080
User-Agent: HTTPie/3.2.1



HTTP/1.1 200
Connection: keep-alive
Content-Length: 13
Content-Type: text/plain;charset=ISO-8859-1
Date: Tue, 11 Apr 2023 08:02:11 GMT
Keep-Alive: timeout=60

Hello Spring
```

## 프론트 컨트롤러

서블릿은 요청마다 매핑이 필요

서블릿이 늘어나고 매핑받고 하다 보니까 여러 서블릿에서 공통적으로 필요한 작업이 각 서블릿 안에 중복되는거임

서블릿이 웹 요청과 응답을 직접적으로 request/ response object를 다뤄줘야하는 방식 -> 자연스럽지 못함

=> 기본적인 서블릿을 가지고 개발하기 어려움이 있음

따라서 프론트 컨트롤러 등장

### 프론트 컨트롤러는

각 서블릿은 url에 맵핑되는데 그러지 말고 여러 서블릿에 등장하는 공통 코드를 제일 앞단에 중앙화된 오브젝트에서 처리하고 다른 오브젝트한테 위임하게 하자는 마인드

특정 로직 수행후에 공통된 후처리 로직이 있다면 그것도 프론트 컨트롤러가 처리하자

서블릿을 이용해서 프론트 컨트롤러를 만드는 것이 유행했다.

인증/ 보안, 다국어 처리 등에 사용되었음

```
package tobyspring.helloboot;

import org.apache.catalina.startup.Tomcat;
import org.springframework.boot.SpringApplication;
import org.springframework.boot.autoconfigure.SpringBootApplication;
import org.springframework.boot.web.embedded.tomcat.TomcatServletWebServerFactory;
import org.springframework.boot.web.server.WebServer;
import org.springframework.boot.web.servlet.ServletContextInitializer;
import org.springframework.boot.web.servlet.server.ServletWebServerFactory;
import org.springframework.http.HttpHeaders;
import org.springframework.http.HttpMethod;
import org.springframework.http.HttpStatus;
import org.springframework.http.MediaType;

import javax.servlet.ServletContext;
import javax.servlet.ServletException;
import javax.servlet.http.HttpServlet;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;
import java.io.IOException;



public class HellobootApplication {

        public static void main(String[] args) {

                //Tomcat servlet webserver
                //ServletWebServerFactory serverFactory = new TomcatServletWebServerFactory(); ->
                TomcatServletWebServerFactory serverFactory = new TomcatServletWebServerFactory();

                //servlet container
                //servlet container      object
                WebServer webServer = serverFactory.getWebServer(servletContext -> {
                        servletContext.addServlet("frontController", new HttpServlet() {
                                @Override
                                protected void service(HttpServletRequest req, HttpServletResponse resp) throws
ServletException, IOException {
                                        //    ,

                                        //  front controller servlet     ( servlet container )
                                        if(req.getRequestURI().equals("/hello") && req.getMethod().equals
(HttpMethod.GET.name())){ //
                                                String name = req.getParameter("name");

                                                resp.setStatus(HttpStatus.OK.value());
                                                resp.setHeader(HttpHeaders.CONTENT_TYPE, MediaType.
TEXT_PLAIN_VALUE);

                                                resp.getWriter().println("Hello " + name); //getWriter.print:
object to string .
                                        } else if (req.getRequestURI().equals("/user")) {
                                                //user
                                        } else{
                                                resp.setStatus(HttpStatus.NOT_FOUND.value());
                                        }
                                }
                        }).addMapping("/*"); // /        -> front controller
                });

                // Tomcat servlet container
                webServer.start();
        }

}
```

```
 http -v ":8080/hello?name=Spring"
GET /hello?name=Spring HTTP/1.1
Accept: */*
Accept-Encoding: gzip, deflate
Connection: keep-alive
Host: localhost:8080
User-Agent: HTTPie/3.2.1



HTTP/1.1 200
Connection: keep-alive
Content-Length: 13
Content-Type: text/plain;charset=ISO-8859-1
Date: Tue, 11 Apr 2023 08:21:03 GMT
Keep-Alive: timeout=60

Hello Spring


 http -v POST ":8080/hello?name=Spring"
POST /hello?name=Spring HTTP/1.1
Accept: */*
Accept-Encoding: gzip, deflate
Connection: keep-alive
Content-Length: 0
Host: localhost:8080
User-Agent: HTTPie/3.2.1



HTTP/1.1 404
Connection: keep-alive
Content-Length: 0
Date: Tue, 11 Apr 2023 08:21:16 GMT
Keep-Alive: timeout=60
```

## Hello 컨트롤러 매핑과 바인딩

### mapping

웹 요청에 들어있는 정보를 활용해서 어떤 로직을 수행하는 코드를 실행할 것인가 결정하는 작업

### binding

~~웹 요청이나 응답을 직접 다루는 오브젝트를 사용하지 않고 로직을 분리하는 것 (helloController)~~

직접적으로 웹 요청/응답 다루는 오브젝트 사용하지 않고, 평범한 자바 타입으로 (웹 요청) 정보를 변환해서 사용

평범한 java 타입으로 변환된 요청 정보를 받아서 로직을 수행함

웹 요청의 값들을 java 타입으로 바꿔서 메소드를 호출할 때 인자값으로 넘겨주는 작업 == binding

```
package tobyspring.helloboot;

import org.apache.catalina.startup.Tomcat;
import org.springframework.boot.SpringApplication;
import org.springframework.boot.autoconfigure.SpringBootApplication;
import org.springframework.boot.web.embedded.tomcat.TomcatServletWebServerFactory;
import org.springframework.boot.web.server.WebServer;
import org.springframework.boot.web.servlet.ServletContextInitializer;
import org.springframework.boot.web.servlet.server.ServletWebServerFactory;
import org.springframework.http.HttpHeaders;
import org.springframework.http.HttpMethod;
import org.springframework.http.HttpStatus;
import org.springframework.http.MediaType;

import javax.servlet.ServletContext;
import javax.servlet.ServletException;
import javax.servlet.http.HttpServlet;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;
import java.io.IOException;



public class HellobootApplication {

        public static void main(String[] args) {

                //Tomcat servlet webserver
                //ServletWebServerFactory serverFactory = new TomcatServletWebServerFactory(); ->
                TomcatServletWebServerFactory serverFactory = new TomcatServletWebServerFactory();

                //servlet container
                //servlet container      object
                WebServer webServer = serverFactory.getWebServer(servletContext -> {

                        //
                        HelloController helloController = new HelloController();

                        servletContext.addServlet("frontController", new HttpServlet() {
                                @Override
                                protected void service(HttpServletRequest req, HttpServletResponse resp) throws
ServletException, IOException {
                                        //    ,

                                        // front controller servlet     ( servlet container )
                                        if(req.getRequestURI().equals("/hello") && req.getMethod().equals
(HttpMethod.GET.name())){ // mapping
                                                // http request
                                                String name = req.getParameter("name");

                                                //  (  )
                                                String ret = helloController.hello(name);

                                                resp.setStatus(HttpStatus.OK.value());
                                                resp.setHeader(HttpHeaders.CONTENT_TYPE, MediaType.
TEXT_PLAIN_VALUE);

                                                resp.getWriter().println(ret); //getWriter.print: object to
string .
                                        }
                                        else if (req.getRequestURI().equals("/user")) {
                                                //user
                                        }
                                        else{
                                                resp.setStatus(HttpStatus.NOT_FOUND.value());
                                        }
                                }
                        }).addMapping("/*"); // /        -> front controller
                });

                // Tomcat servlet container
                webServer.start();
        }

}
```