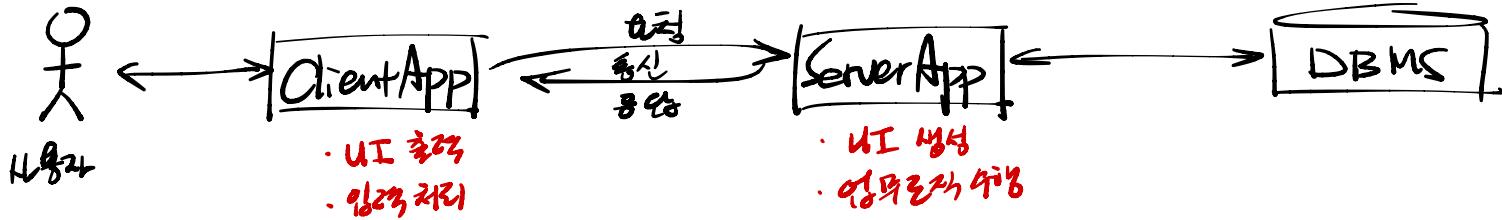
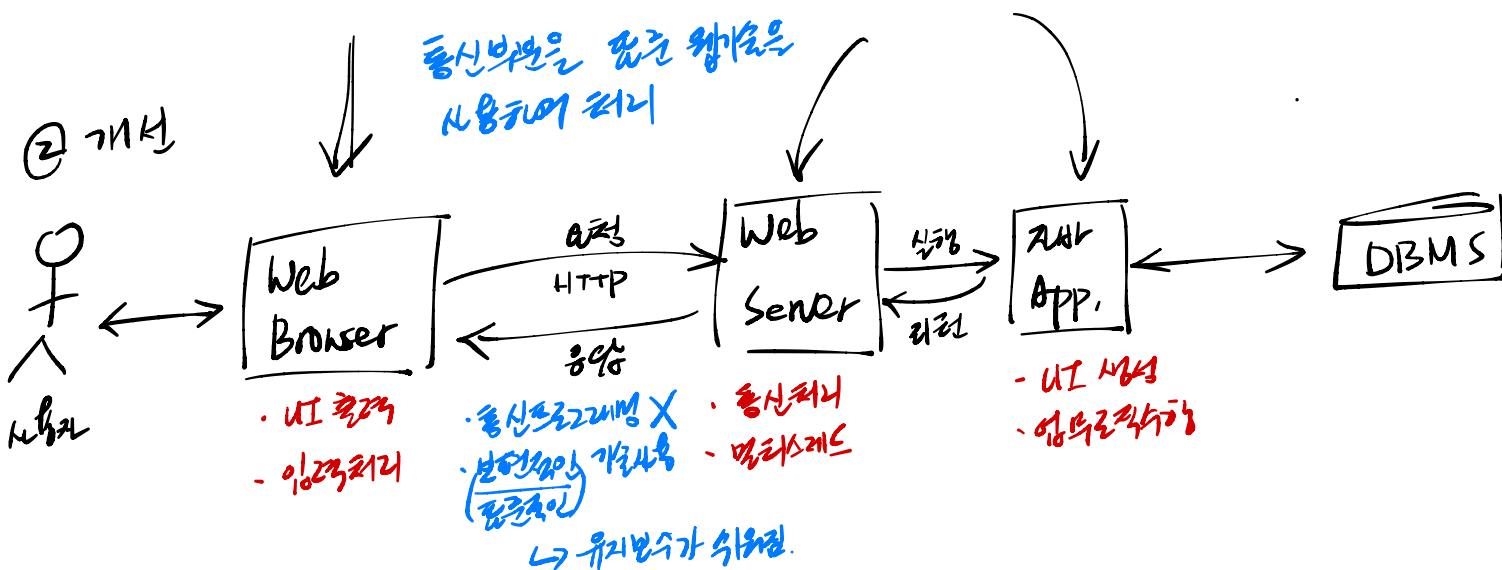


## 49. 웹애플리케이션 구조를 살펴보자

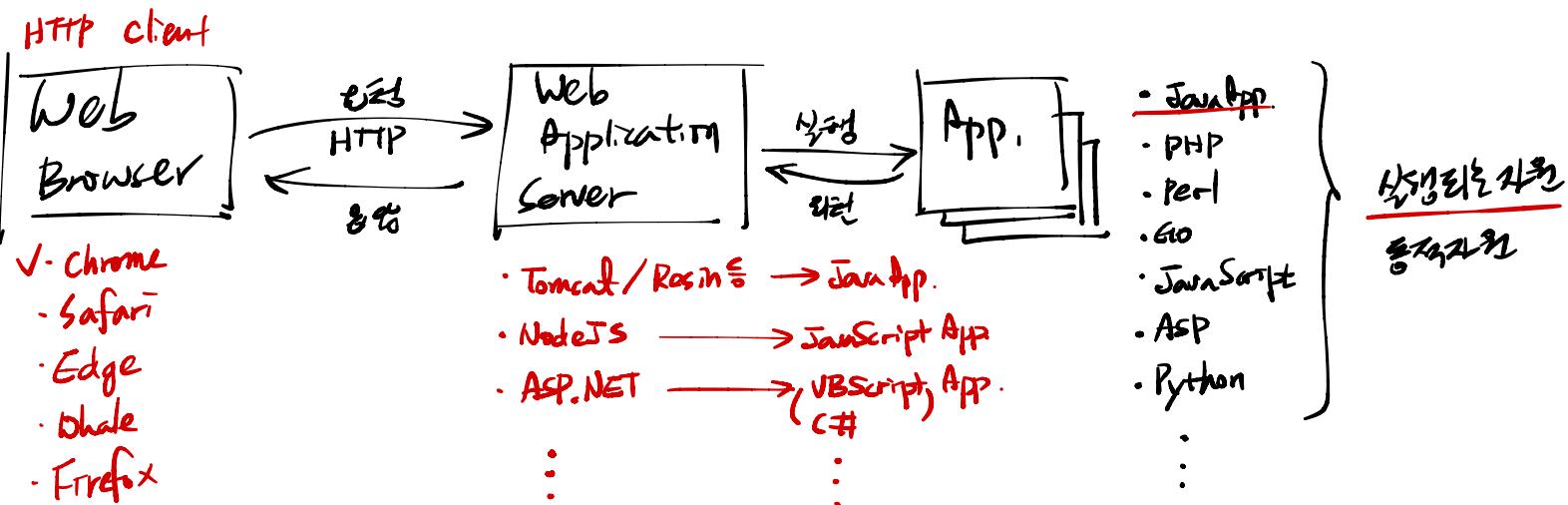
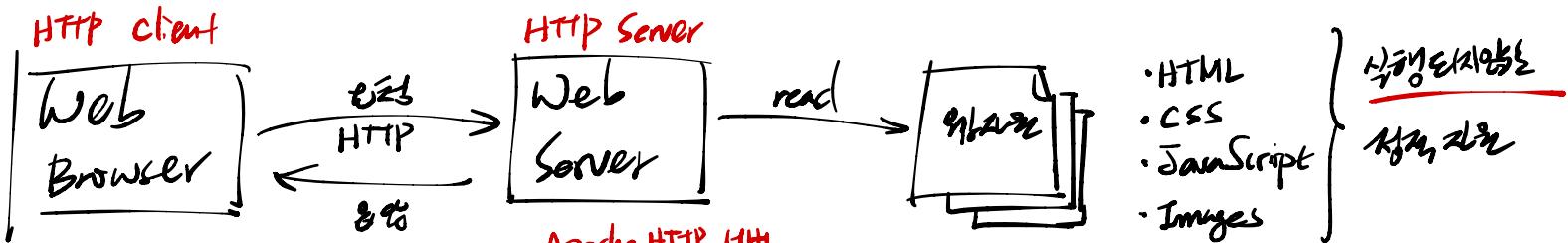
① 단계 → 전용 프로토콜을 사용하여 클라이언트/서버 통신



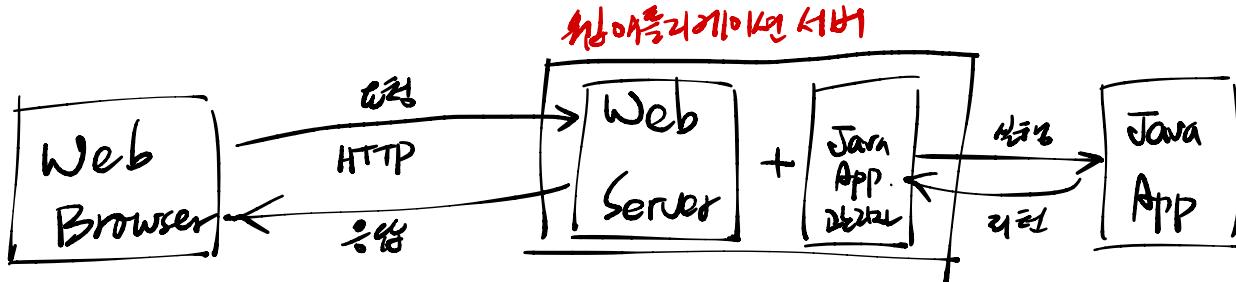
② 개발



\* 웹 사이트의 구조와 작동 원리

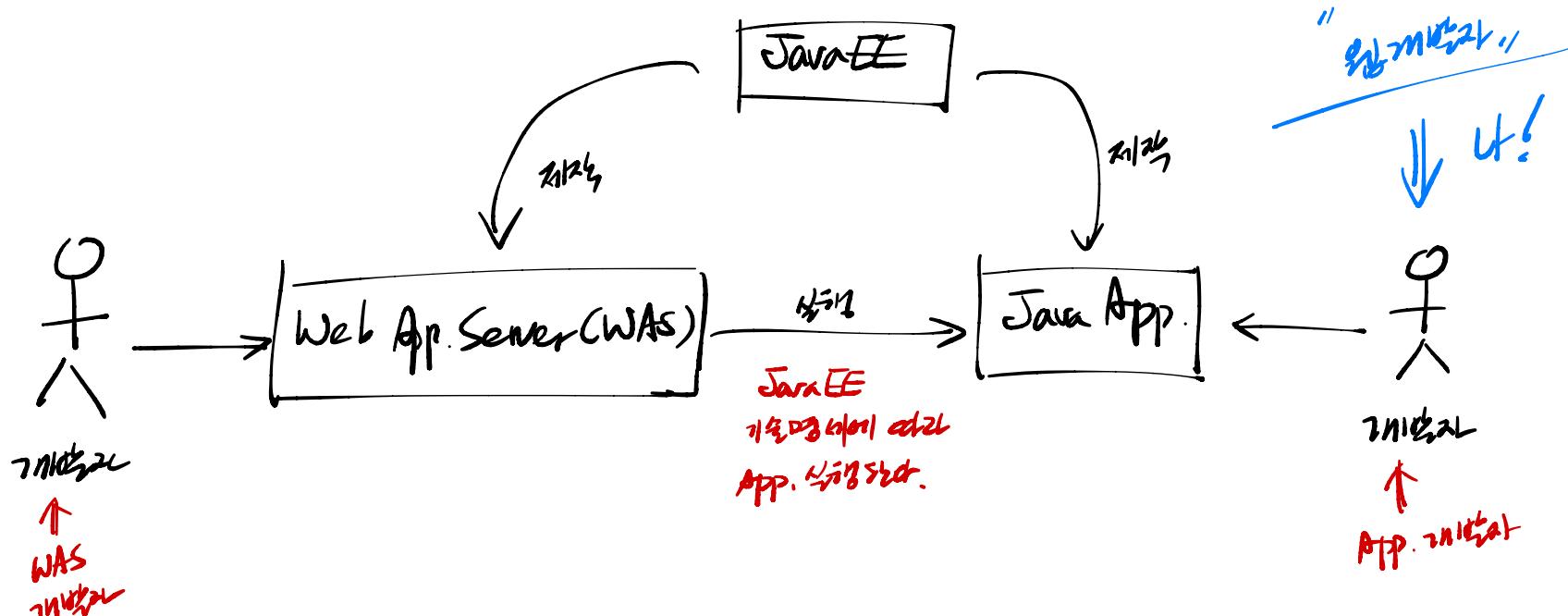


# \* Java Web Application Architecture



- Tomcat / Resin / Jetty : ~~轻量级~~
  - Weblogic
  - Websphere
  - JBoss
  - BEA
  - IBM
  - JEUS
  - :

## \* Java Web Application ut JavaEE چیぞ咯!



\* Java EE (Enterprise Edition) 기술 영역

↳ 기업용 App. 개발 기술 모음

↳ 데이터베이스, 공유자원 관리, 분산처리, 접근제한 등

애플리케이션

Web



Servlet/JSP

JSTL > EL

ESB

Web Service

Authentication  
&  
Deployment

- 웹 애플리케이션 개발 기술

← 핵심 기술!

⇒ REST API

- 분산 처리 기술

- 서비스 기반으로 서비스 개발

- 인증 및 배포 기술

## \* Java EE 7/8/10/11 버전의 특징

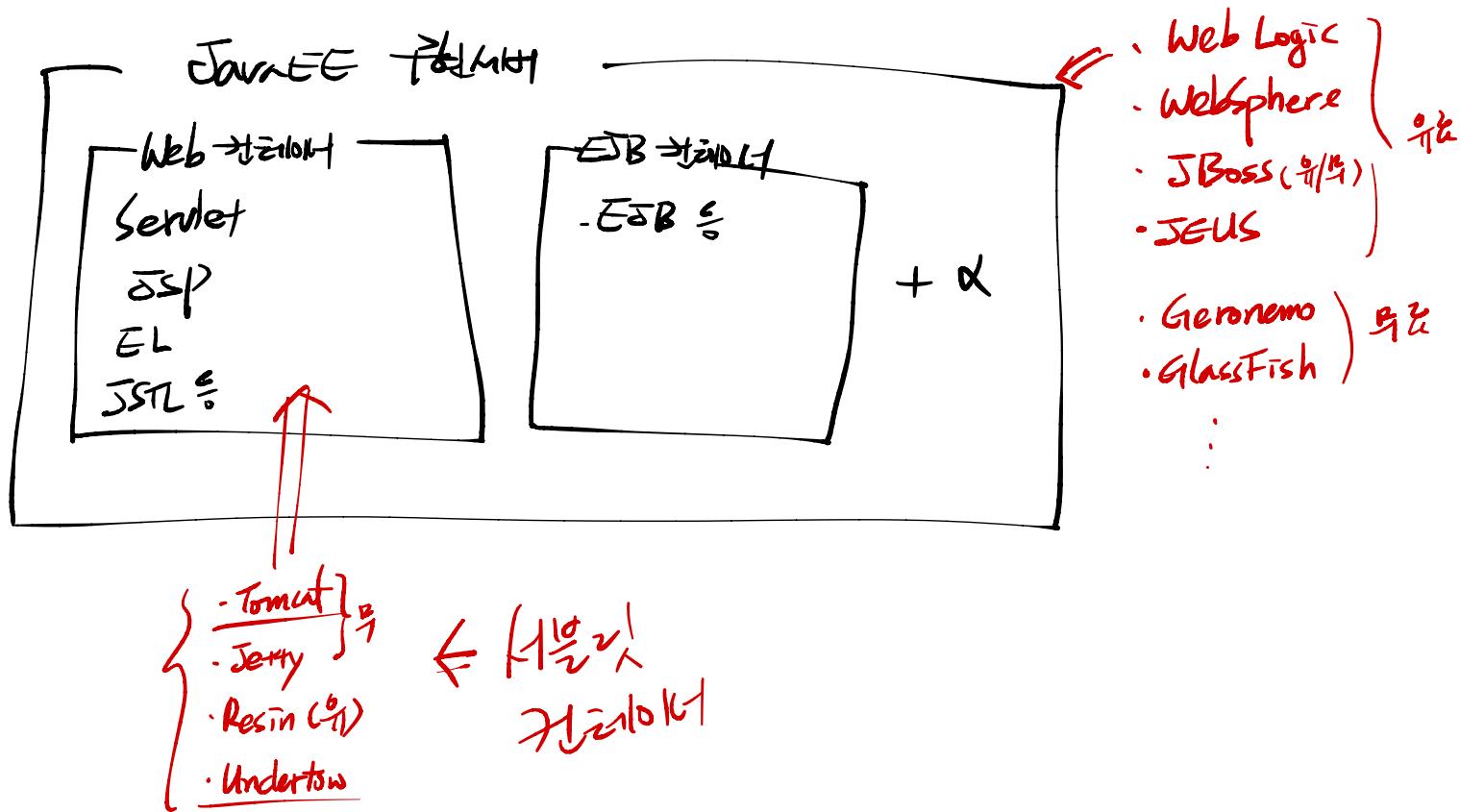
<del>JavaEE 버전</del> <del>제작년도</del>	Servlet	JSP	EL	EB
5	2.5	2.1		3.0
6	3.0	2.2	2.2	3.1
7	3.1	2.3	3.0	3.2
8	4.0	2.3	3.0	3.2

## \* Java EE 7/8 버전별 지원 현황

<del>JavaEE 6 버전</del>	Web Logic	Web Sphere	JBoss	Tomcat
5	10.3	6.1, 7.0	6.0	6.0
6	11g, 12c(12.1.x)	7.0, 8.0	7.0	7.0
7	12c(12.1.3), 12.2.x	8.5, 9.0	8.0	8.0
8	12.2.1, 14.1.x	9.0.x	9.1, 8.2	8.5, 9.x

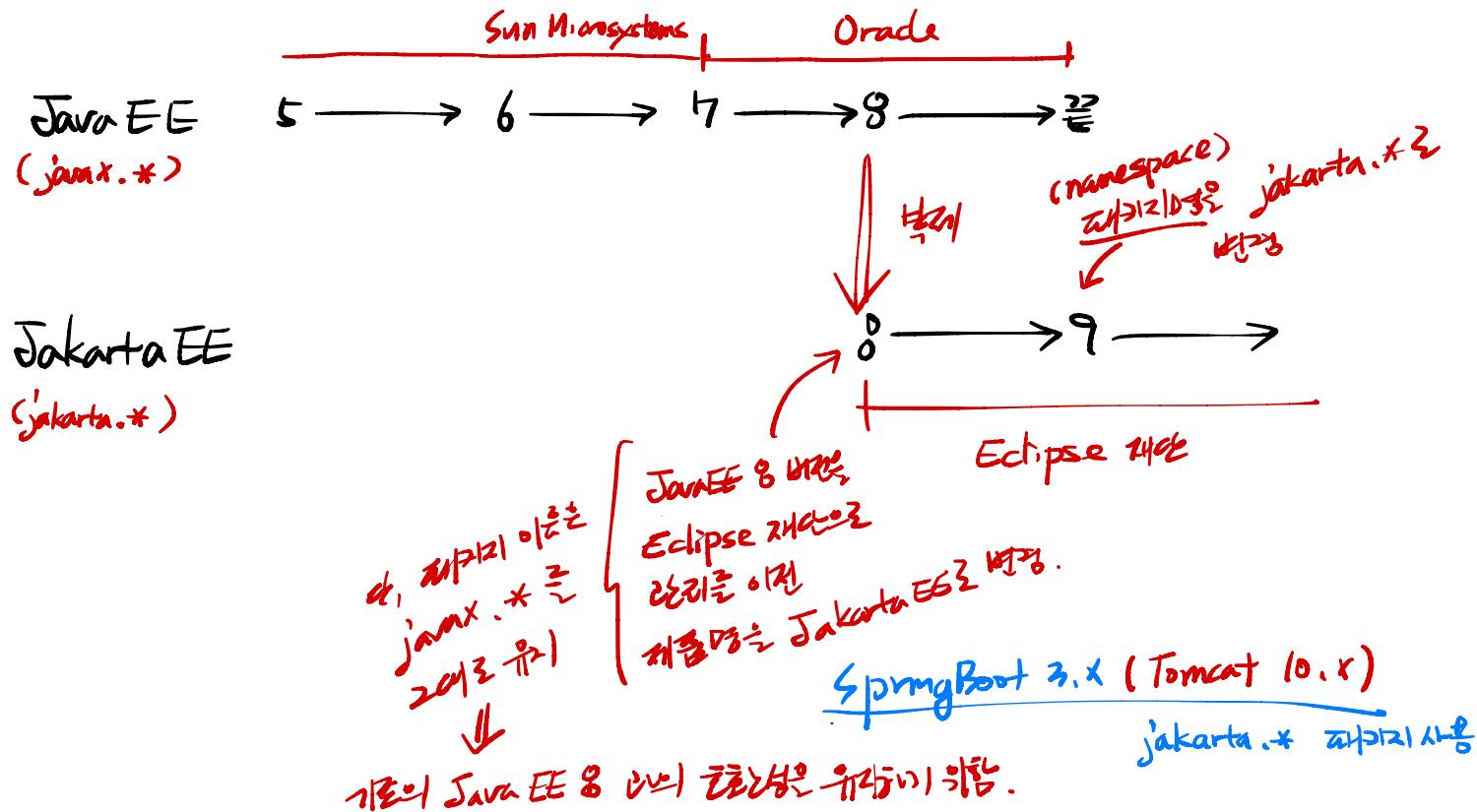
⇒ 2014년 10월 버전으로  
2014년 Java EE 버전별 지원 현황은 다음과 같다.

## \* JavaEE 페미터 서브 분류

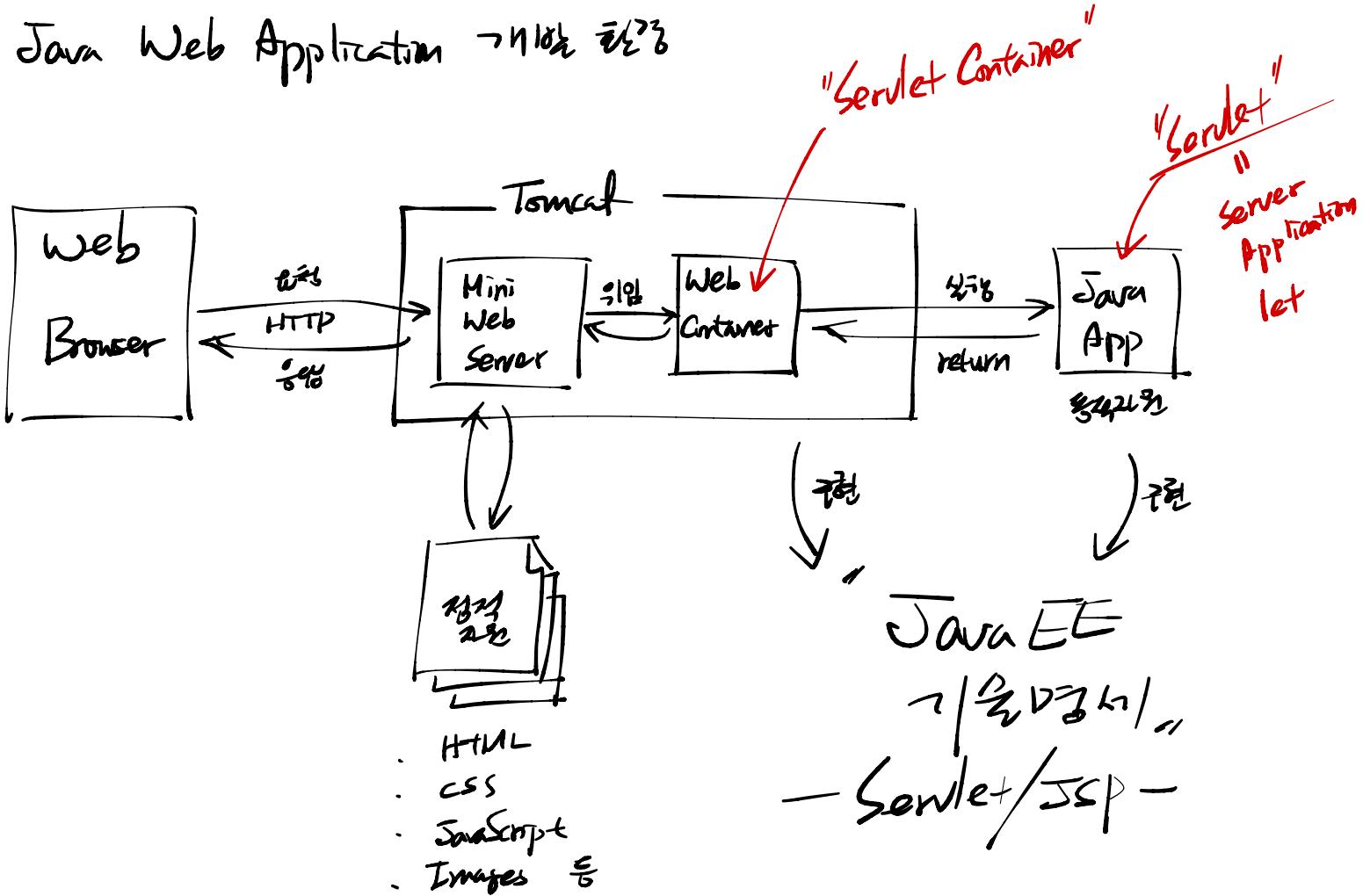


## \* JavaEE & JakartaEE

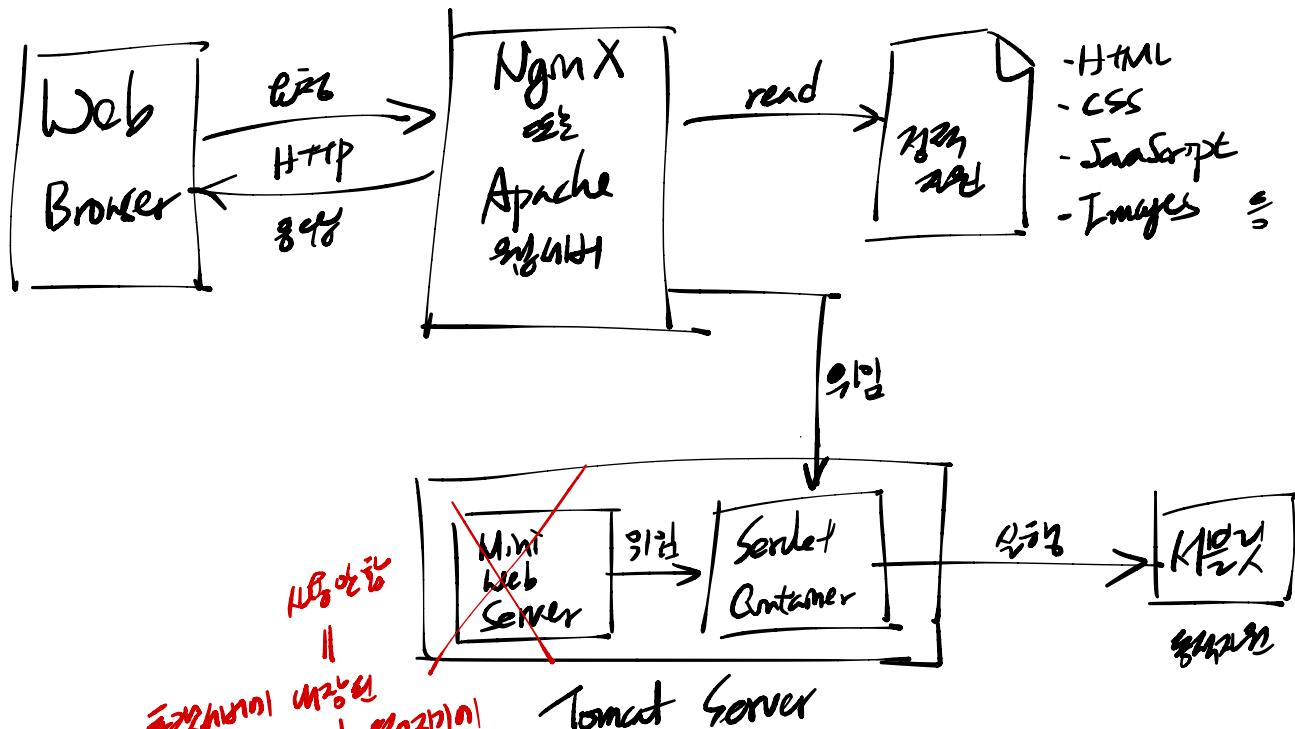
SpringBoot 2.x.x (Tomcat 9.x)



\* Java Web Application → چی یعنی



## \* Java Web Application 운영 원리



External module  
application layer  
middle layer  
presentation layer  
business logic  
data layer.

## \* Tomcat Server

CATALINA\_HOME/

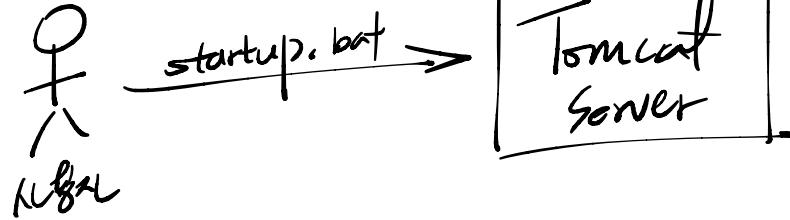
- bin/ ← 허버 실행과 관련된 파일
- conf/ ← 허버 설정 파일
- lib/ ← 허버 라이브러리 파일
- logs/ ← 허버 실행 로그 파일
- temp/ ← 허버 실행 중에 사용하는 파일을 두는 폴더
- webapps/ ← 웹 어플리케이션 폴더를 두다.
- work/ ← JSP를 Servlet 처리하기로 변환한 파일을 두는 폴더

启动文件 => startup.bat (Windows)  
startup.sh (Mac, Linux)

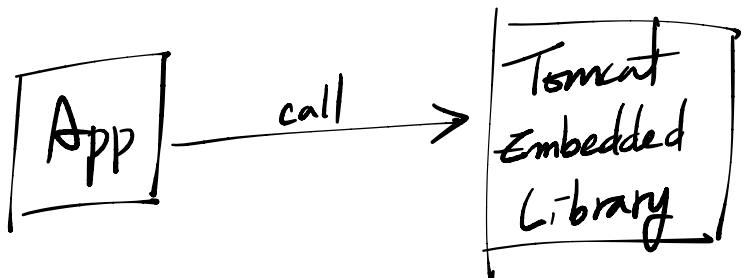
停止文件 => shutdown.bat (Windows)  
shutdown.sh (Mac, Linux)

## \* Tomcat 끊기 방식

① 재시작



② 리부트



## \* Embedded Tomcat 끝까지 써보기

new Tomcat()

- port = 8888
- baseDir = temp
- URLEncoder = UTF-8

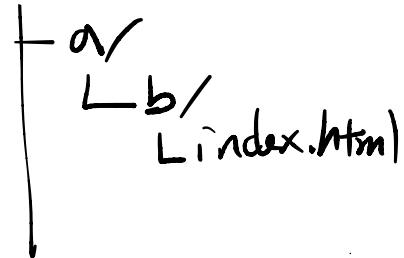
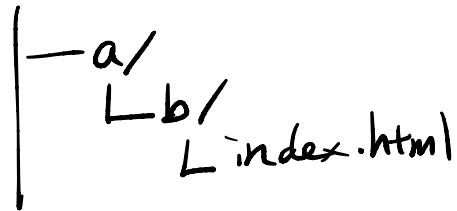
- 웹 어플리케이션 경로  $\Rightarrow$  /
  - 정적 자원 경로  $\rightarrow$  src/main/webapp/  $\leftarrow$  .html, .css, .js, .gif 등
  - 동적 자원 경로  $\rightarrow$  bin/main  $\leftarrow$  .class, .properties, .xml 등

웹 자원 주소: http://localhost:8888 ↗

- ① 정적 파일 경로인 /index.html 을 찾는다.
- ② 동적 파일 경로인 /index.html 3 번째가 실행된 내용을 찾는다.

\* აბსოლუტური

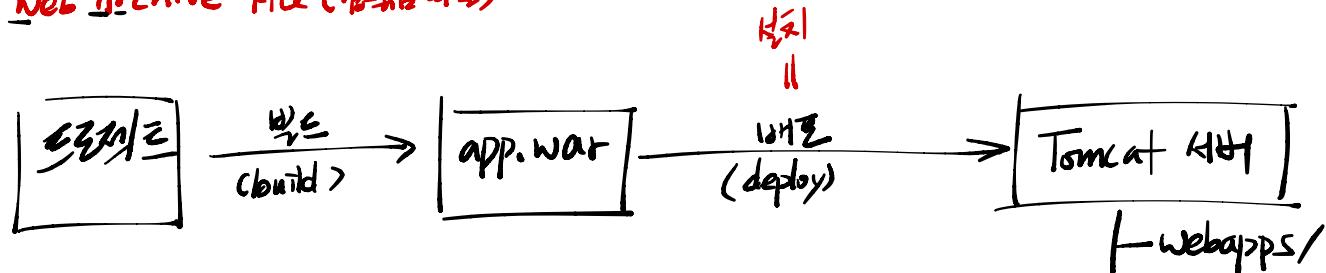
http://localhost:8888/ → src/main/webapp/



\* 웹 어플리케이션 프로젝트 배포

① .war 파일 배포

Web Archive File (웹아카이브)



② 코드 + API 라이브러리 (라이브러리)  
= springboot 앱



\* Maven 파일은 쉽게 편리한 확장자를 가짐

구조도



.html, .css, .js, .gif 등 이미지 파일

WEB-INF

web.xml ← 딜레이션 데스크립터 (Deployment Descriptor File; DD File)

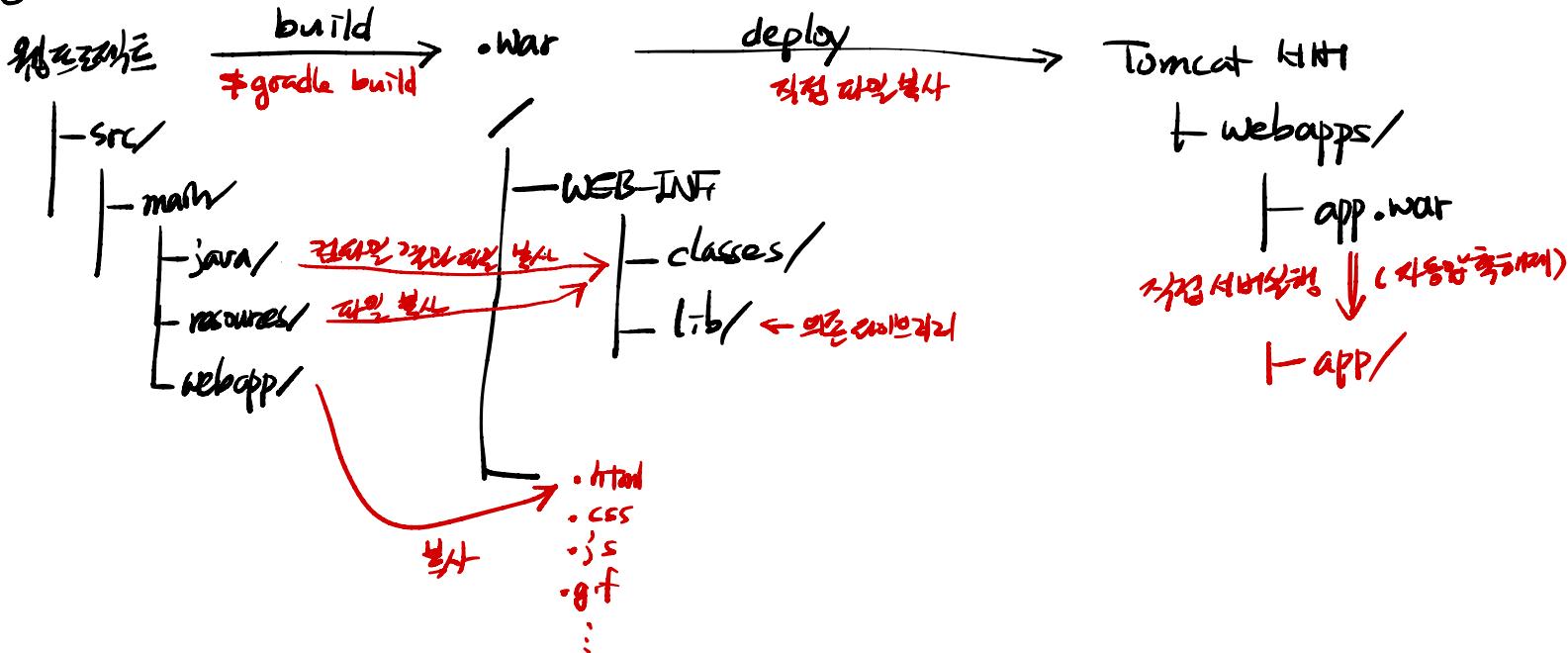
classes/ ← .class, .properties, .xml

lib/ ← .jar

.xml, .properties 등 설정 파일

\* 웹프로젝트 디렉토리

① 디렉토리 구조  $\leftarrow$  웹애플리케이션 기본을 갖춘 후에 버전으로 1842



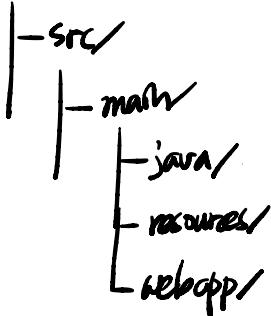
\* 웹프로젝트 파일

② Eclipse 웹 파일 <서버 설정을 적용하는 동안 서버를 파일에

생성되는

Eclipse 임시 파일 폴더

이경스 위치:/.metadata/.plugins/org.eclipse.wst.server.core/



tmp0/

- conf/ ← tomcat 설정 /xml, 폴더는 복사됨

- logs/ ← 실행 기록 파일

- temp/ ← 일정 주기로 자동으로 삭제되는 폴더

- webapps/ ← 새롭게 만든

- work/ ← JSP 변환 파일 생성되는 폴더

- wtpwebapps/ ← 웹프로젝트 파일 폴더

생성되는/

- WEB-INF  
  |  
  + classes/  
  |  
  + lib/

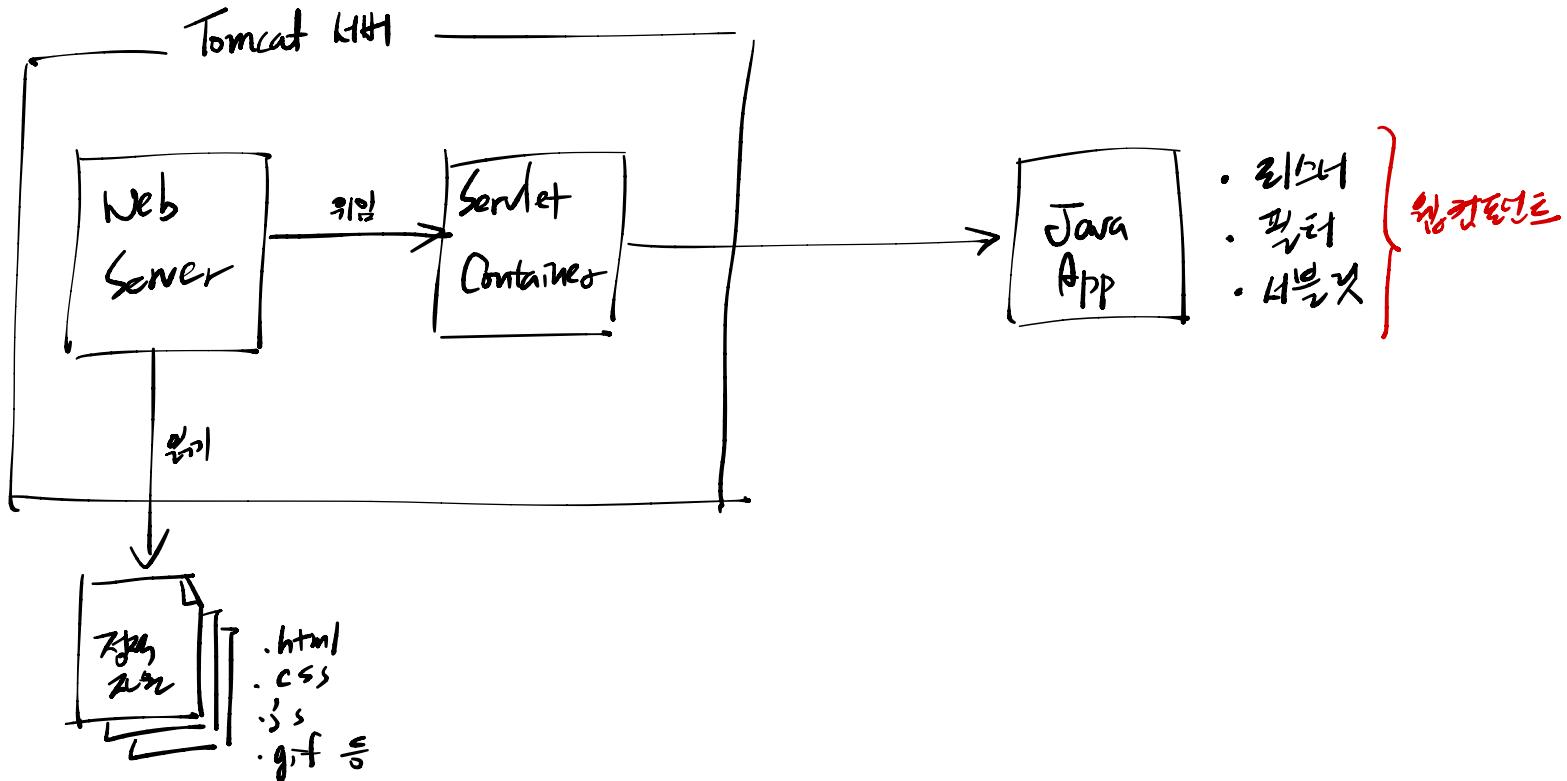
생성되는

\* 웹프로젝트 ID IntelliJ

② 웹프로젝트 + WAS 라이브리 <= SpringBoot 환경



\* 웹 컨테이너(부록)  
↳ 한 개 이상의 기능으로 구성되어 특별한 권한을 부여받을 것. = 특별한 역할은 부여받는 기능



\* 8.7장 틀 번역 - 31/54 = Observer  
Grati

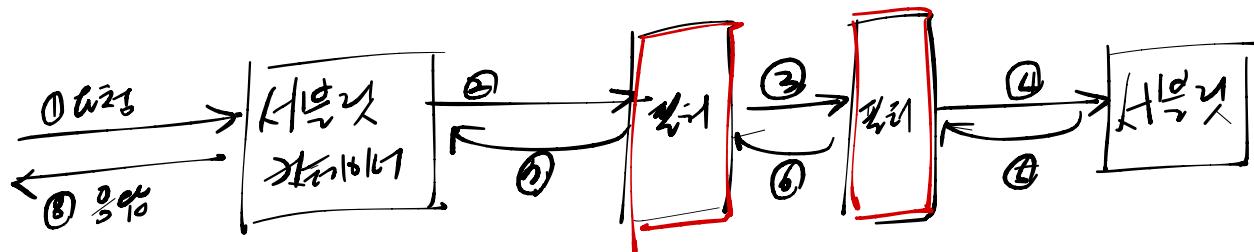
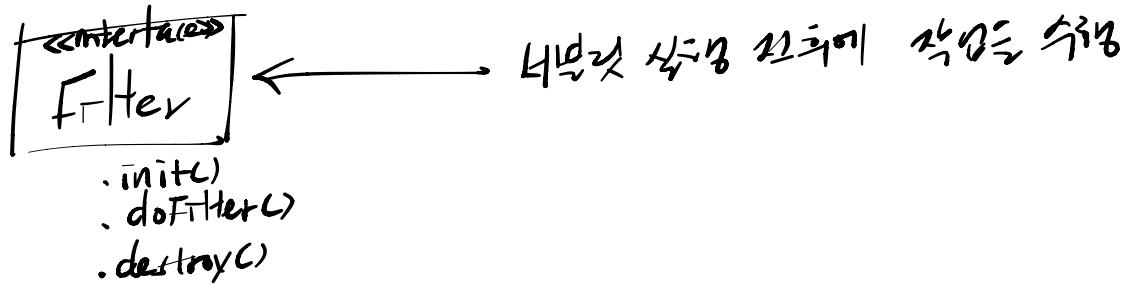
ServletContextListener ← 서버가 컨텍스트가 제작되거나 종료될 때 호출되는  
contextInitialized() contextDestroyed()

ServletRequestListener ← 요청이 제작되거나 제거될 때 호출되는  
requestInitialized() requestDestroyed()

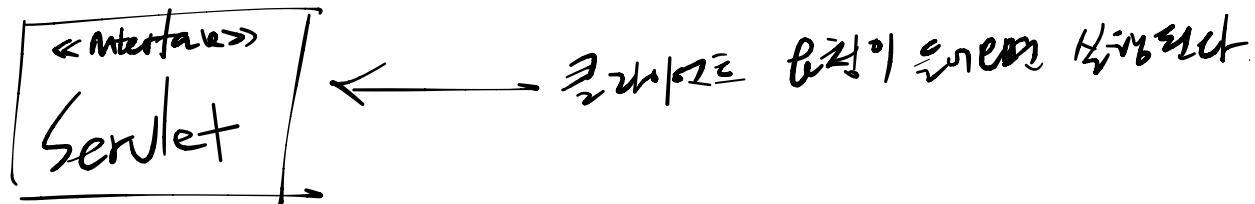
HttpSessionListener ← 클라이언트가 세션을 만들 때 또는 세션을 종료할 때  
sessionCreated() sessionDestroyed()

:

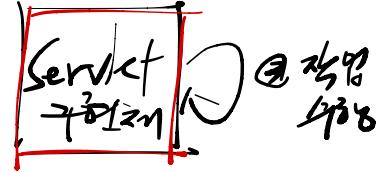
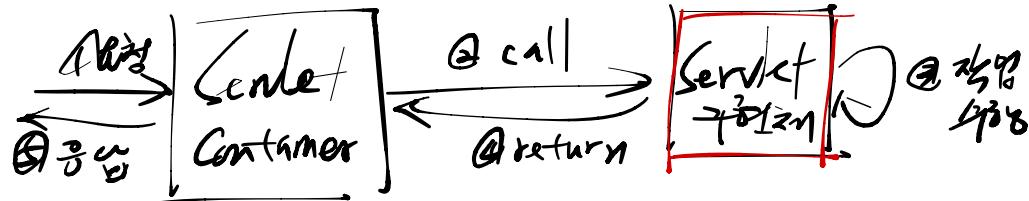
\* 필터링 책임 - 책임 = chain of Responsibility  
 GOF의 책임 전달



\* 웹 개발언어 - HTML = Command  
 GOF의 디자인 패턴



- . init()
- service()
- . destroy()
- . getServletInfo()
- . getServletConfig()



\* 풀不尽 서버로 진화하여 등장

① 웹서버 → ② 애플리케이션 서버

Server App

- 파일통신구현
- 서비스레이팅 구현
- 전송통신망
- 웹서버 구현

Server App

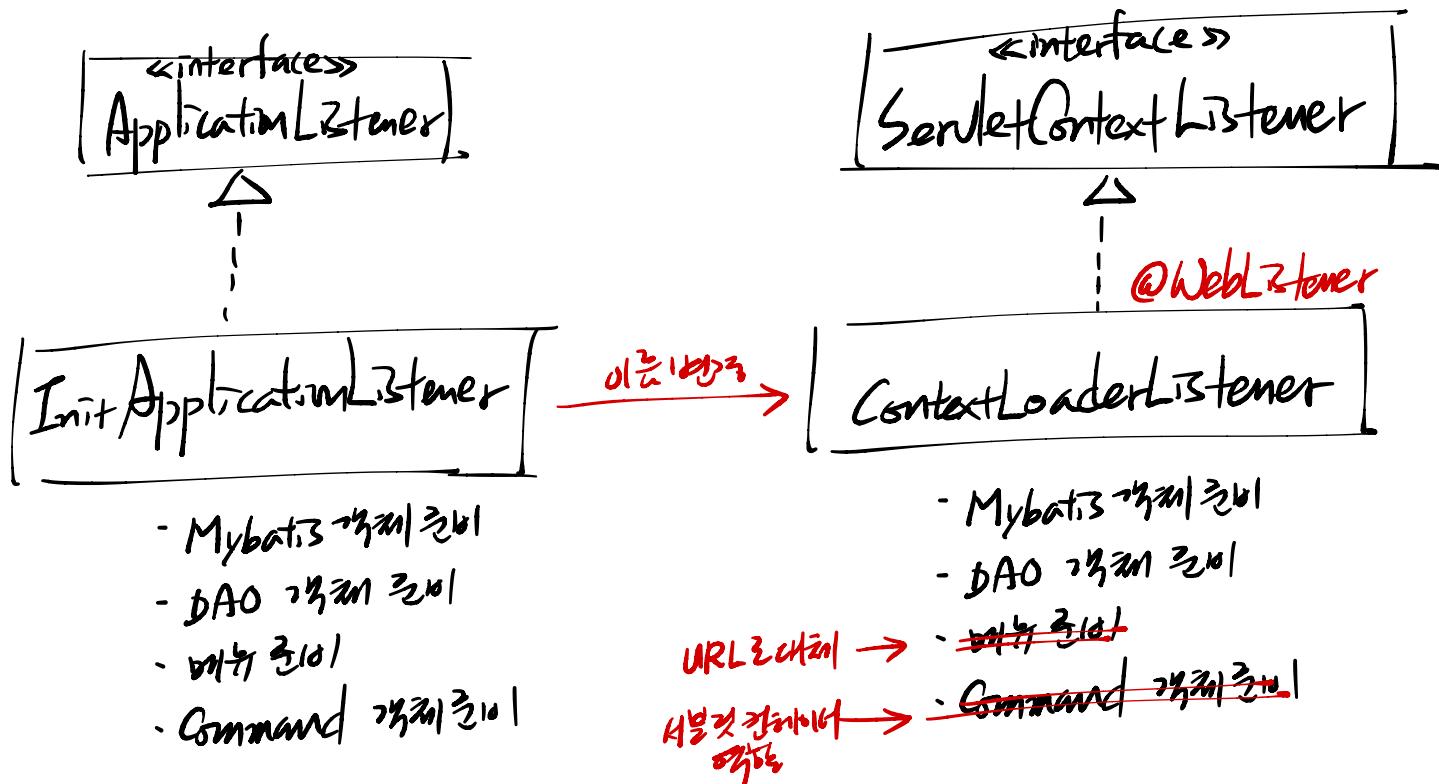
- ~~파일통신구현~~
- ~~서비스레이팅 구현~~
- ~~전송통신망~~
- ~~웹서버 구현~~

Web

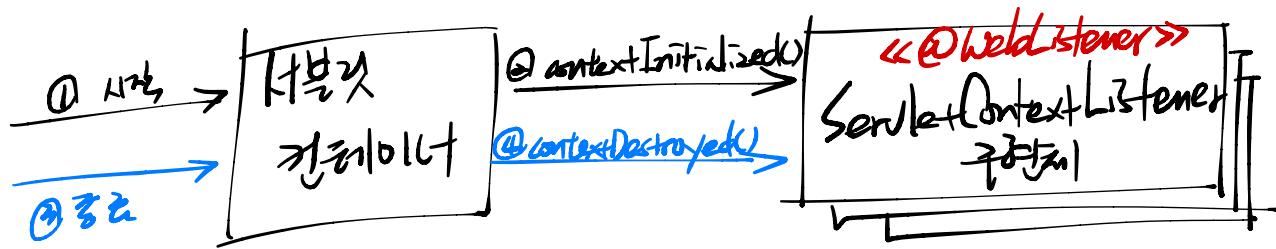
Tomcat

- HTTP 프로토콜을 처리하는 웹브라우저 구현
- 서비스레이팅 구현
- 전송통신망 구현
- 웹서버 기능으로 구현
- 웹사이트의 URL을 기능 구현

\* 리스너 만들기



## \* ServletContext Listener

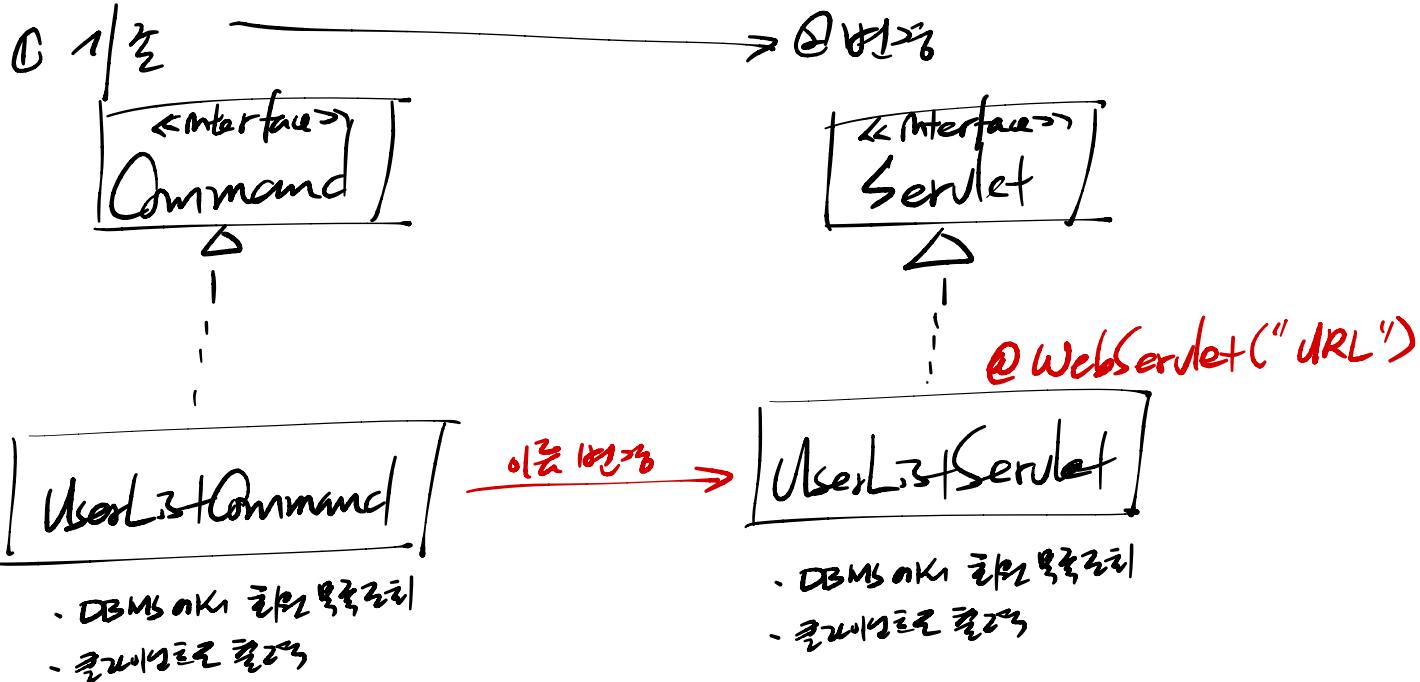


\* **@WebListener** effect하기

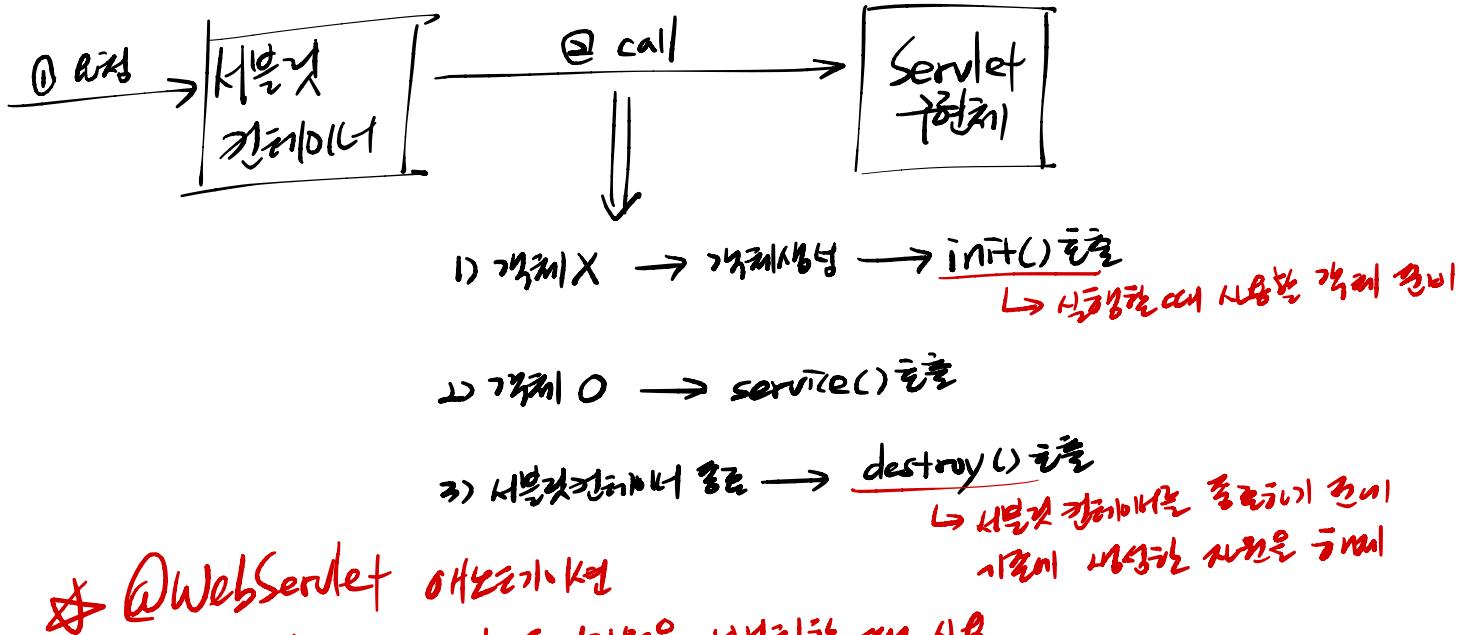
- 라이브러리에 수동적 컨테이너에 등록하는 방법

수동적 컨테이너에 등록하는 방법

\* 허용 가능한 방법



## \* Servlet 구조



## \* WebServlet의 특징

- 서블릿작동이 끝나면 종료되는 대체로 특징은 유지된다.
- 클라이언트에게 응답을 때 그 내용을 처리하는 코드를 작성한다.
- URL을 이용하여 코드를 처리하는 처리를 적용한다.

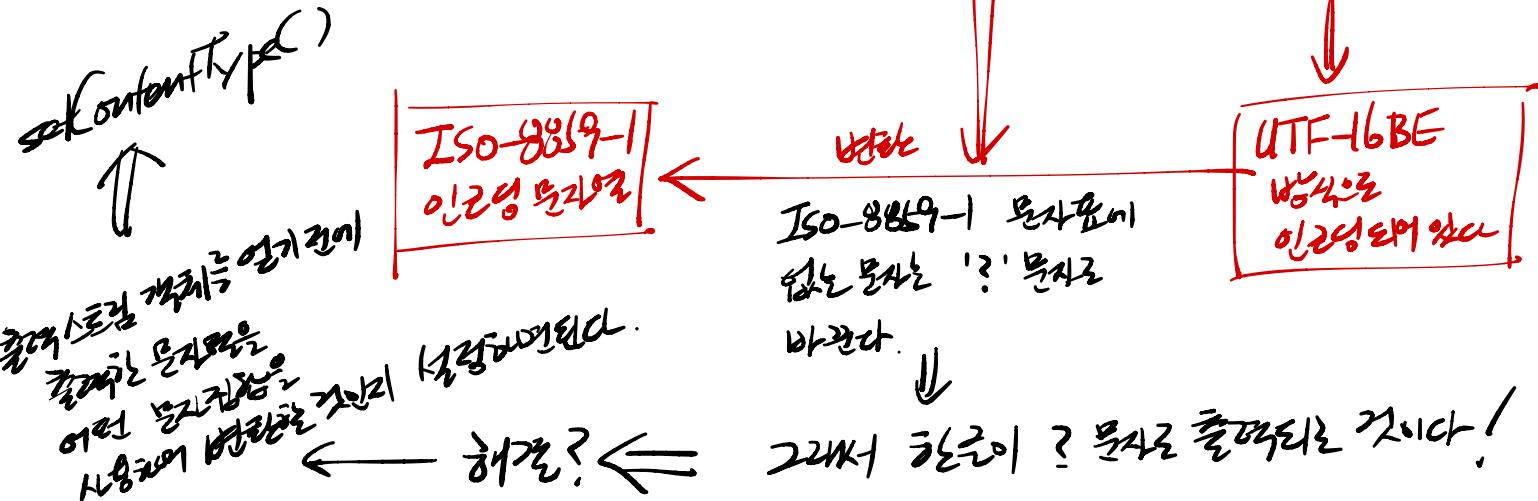
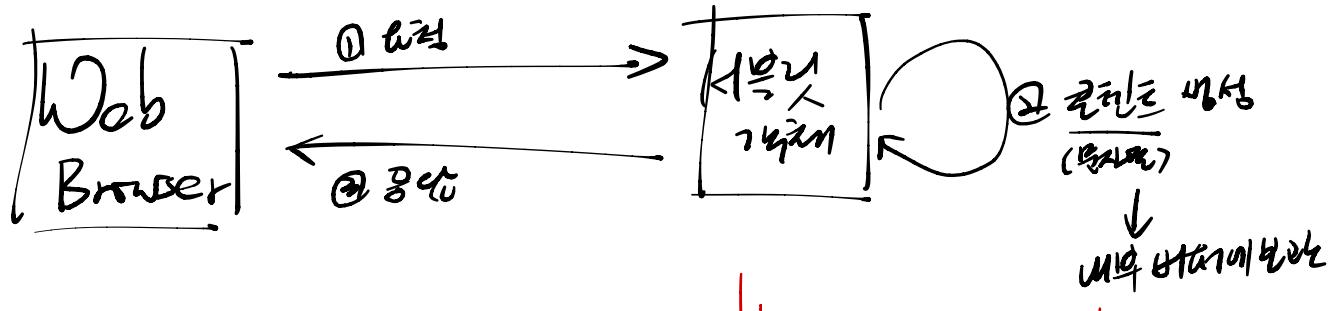
\* service() method

service(ServletRequest req, ServletResponse resp) { }

- ✓ Request pass thru
- ✓ Response thru Method call

• Request thru Method call

\* 문자열 출력하고 읽을 때 깨짐



\* 한글이 흐赡한 문자열에서 한글이 깨지는 예

한글

"ABC??"



442433F3F

한글

한글

"ABC가?"



004100420043 AC00AC01 (UTF-16BE)

ISO-8859-1 문자셋 (256자)

A → 41  
B → 42

a → 61  
b → 62

o → 30  
i → 31

:

\* UTF-16BE vs UTF-16LE  
" (UTF-16)

'f' 0xAC00 0x00AC  
'A' 0x0041 0x4100

\* **내부기능**이 **특정한 동작방법**과 **방법**의 **개념**을 예  $\Rightarrow$  **내부기능**

월정으로 놀러

"ABC→P<sub>L</sub>"

三

~~4443 EAB080 EAB081~~

제작자와 내용  
 $\frac{\text{제작자}}{\text{내용}} = \frac{1}{2}$

서별기

"ABC가족"

1

004|004|0043 A00 A01 (UTF-16BE)

UTF-8 한글판

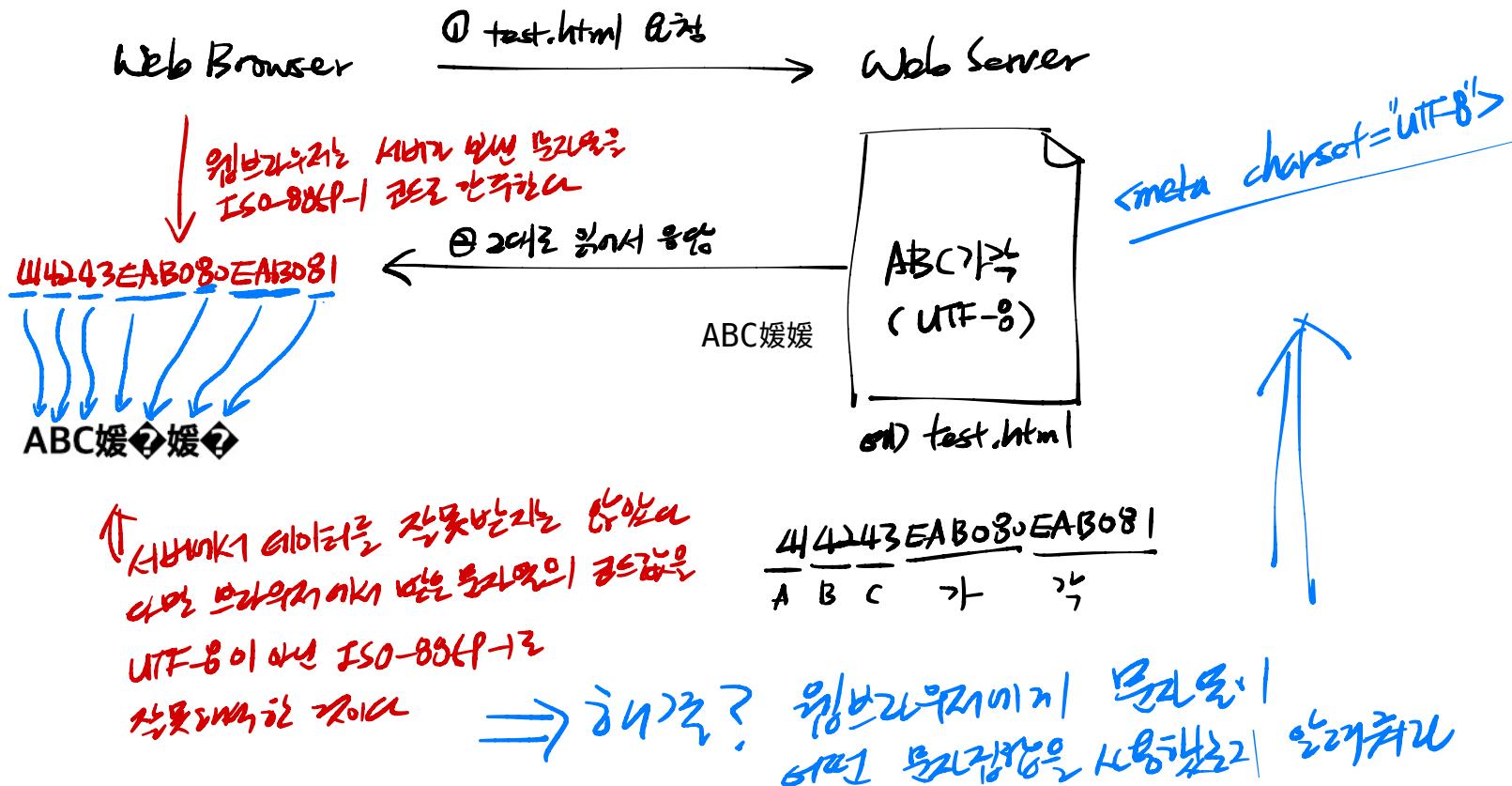
$$\begin{array}{ccc} A & \longrightarrow & 41 \\ B & \longrightarrow & 42 \end{array}$$

1

$$\gamma_L \rightarrow EABO^8.$$

$\rightarrow$  ~~EABO81~~

## \* HTML 문서의 한글 깨짐



\*

\* setContenttype()

Multi-purpose  
Internet  
Mail  
Extension

MIME Type  
설정방법

이메일로 전송되는 경우로  
문서의 확장자와 일치하지 않아  
문서형식  
→ 파일은 미리워크로  
작은 파일로  
파일을 사용하는 경우  
문서형식을 사용하는 경우로  
선택.

setContenttype("MIME 타입; charset");

① MIME의 인터넷 기본

MIME Type 형식

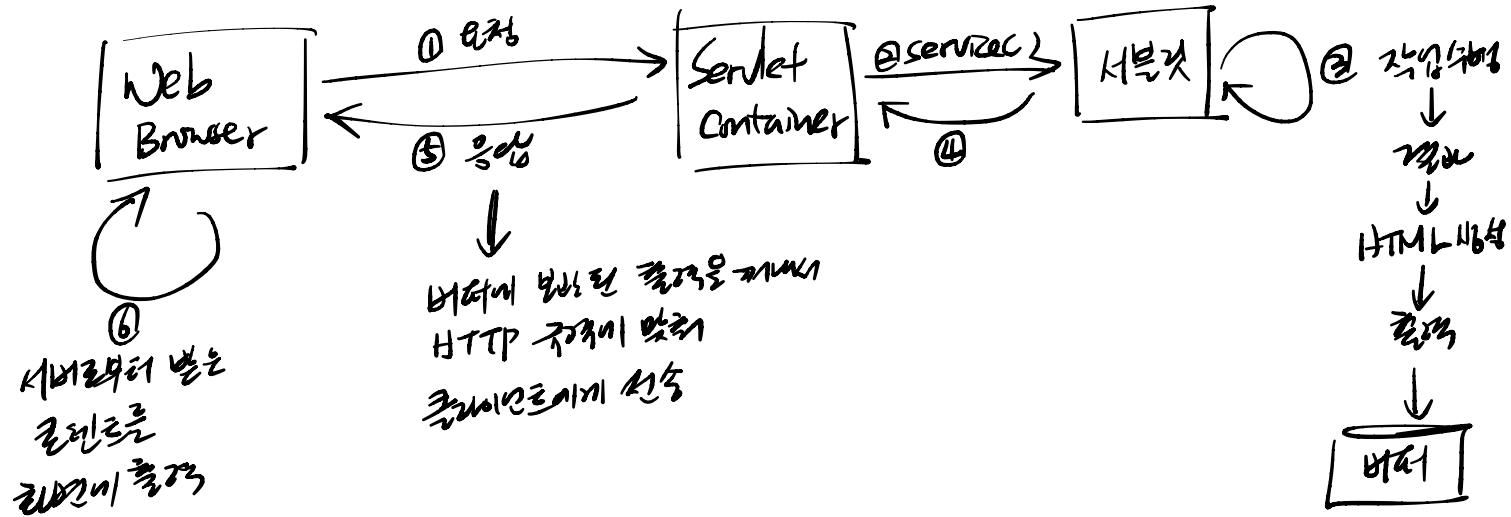
↓  
type/sub-type ; charset = 값

ex) "text/plain; charset=UTF-8"

↓

UTF-16BE → ~~ISO-8859-1~~  
UTF-8

\* Web Browser et HTML  
 ↳ 웹 클라우드를 제공하는 웹서버 = 브라우저 + 웹서버 + 서버 + DB



text/plain → 2013 323

text/html → HTML 렌더링

기타 → 파일

## \* HTML (Hyper-Text Markup Language)

↳ 웹페이지 구조화

