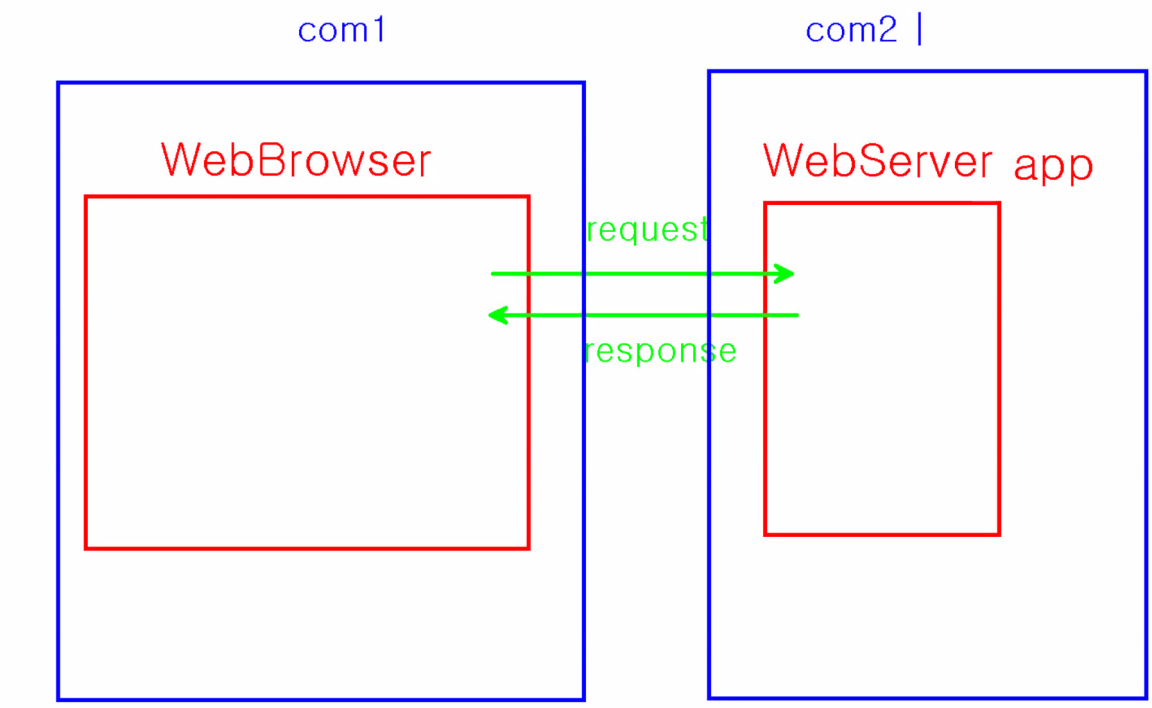
8_10_2022__java_Web

- server 탭에 add and remove로 서버를 불러올 수 있다.
-
- 웹브라우저는 웹서버에서 불러오는 것
- 웹서버는 응답하는 입장



locashost == 127.0.0.1

- webapp 안에 들어있는 데이터들은 공개자원으로 불린다.
  하지만 META-INF, WEB-INF 는 가려지는 폴더(응답 안됨)
  최상위 폴더는 webapp

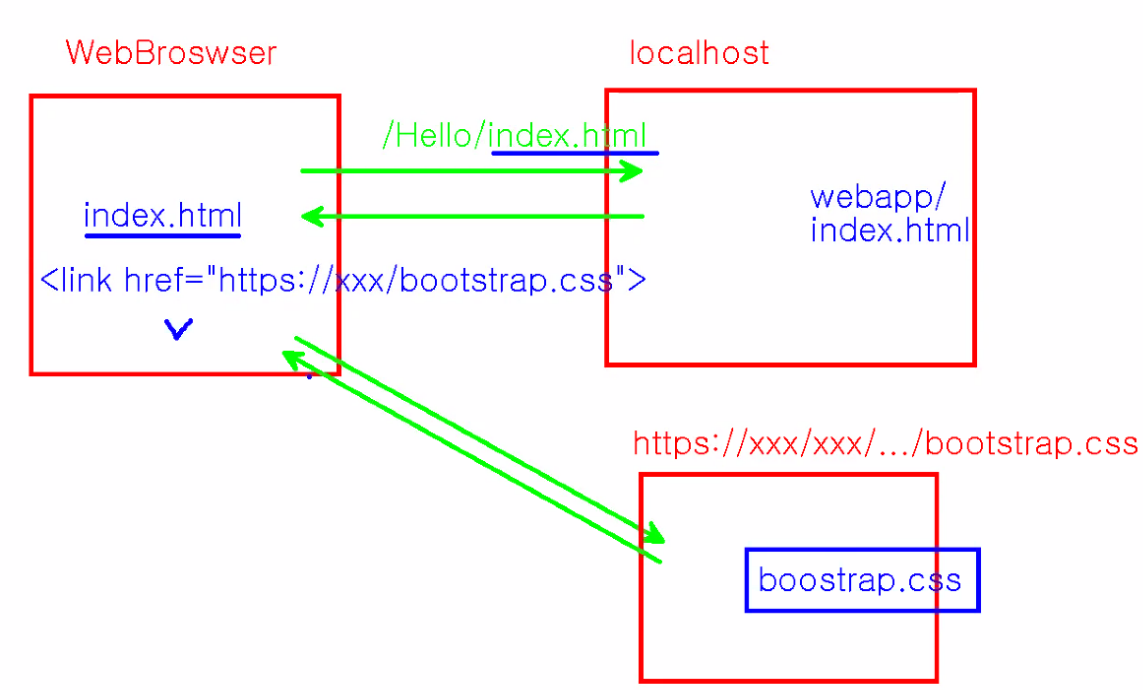
- html요청을 한번 하고 끝내지만 중간에 다시 요청을 할 수도 있다.
  - ex)중간에 이미지 파일을 넣으면 불러오며 다시 로딩함.

- Welcome Page
  - 최상위 경로까지만 입력해도 연결 됨(index.html을 적지 않아도 알아서 접속 됨)
    naver.com/index.html →naver.com/
- bootstrap
  - 링크를 불러와서 로딩할 수 있음

```
<title></title>
<!-- bootstrap css로딩 -->
```

```
<link href="https://cdn.jsdelivr.net/npm/bootstrap@5.2.0/dist/css/bootstrap.min.css" rel="stylesheet" integrity="sha384-gH2yIJqKdN
<script src="https://cdn.jsdelivr.net/npm/bootstrap@5.2.0/dist/js/bootstrap.bundle.min.js" integrity="sha384-A3rJD856KowSb7dwlZdYE
<head>
```

- 웹브라우져에서 불러오는 모양



두개의 서버로부터 불러오는 것

- 퍼블리싱할 때 내 서버 용량에는 link 를 사용하는게 유리하지만 혹시 나중에 어떤 일이 생길 지 모르니까 중요한거라면 개인 서버에 가지고 있는게 좋다.

- 상대경로 이동 ../ 는 상위 폴더를 의미한다. 즉 현재 폴더는 /webapp/sub/ 로 가정 했을 때
  현재 폴더 상위는/webapp/ 폴더가 된다. 따라서 여기서 ../ 는 /webapp/ 폴더를 가리킨다.

실행하는 방법

1. 서버에 프로젝트를 add 해서 서비스 되도록 한다.
2. 서버를 start 한다.
3. 웹브라우저를 실행시킨다.
4. 주소창에 http://localhost:8888/Hello/index.html을 입력해서 웹브라우저가 index.html 페이지를 로딩 하도록 한다.

- or 서버가 꺼져있는 상태에서 run 하기

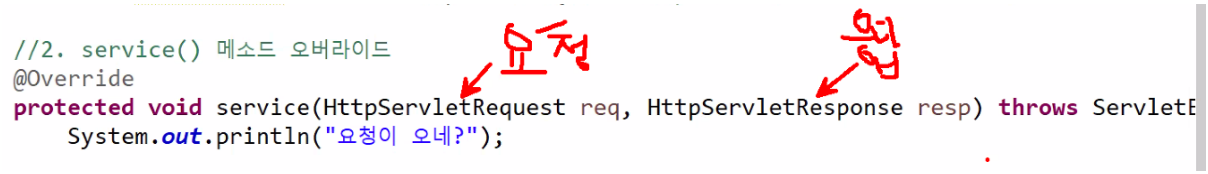
Form 메세지 console 출력

```
<form action="send" method="post">
  <input type="text" name="msg" placeholder="서버에 할말 입력..." />
  <button type="submit">전송</button>
</form>
```

- 오류 : /Hello/send는 가용하지 않습니다.
- 해결 : /send 에 응답할 객체(class)를 만들기

Servlet

- send 라는 Servlet을 만드려 한다.



1. HttpServlet 클래스를 상속받는다

```
public class SendServlet extends HttpServlet
```

2. service() 메소드를 사용하기 위해 오버라이드 한다.

```
@Override
protected void service(HttpServletRequest req, HttpServletResponse resp) throws ServletException, IOException{
  System.out.println("요청이 옵니다.");
}
```

3. 요청 맵핑(처리할 경로) 반드시 /로 시작해야하고 프로젝트명을 쓰지 않는다.

```
@WebServlet("/send")
```

4. 객체 얻어오고 출력 방출 닫기

```
//객체 얻어오기
PrintWriter pw = resp.getWriter();
//웹브라우저에 출력하기
pw.println("Oh my gosh!");
pw.println("what the heck?");
pw.flush();//방출
pw.close();//닫아주기
```

- 완성

```
package test.servlet;

import java.io.IOException;
import java.io.PrintWriter;

import javax.servlet.ServletException;
import javax.servlet.annotation.WebServlet;
import javax.servlet.http.HttpServlet;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;

// 3. 요청 맵핑( 처리할 요청 경로를 작성한다) 반드시 /로 시작을 해야 하고 프로젝트명은 쓰지 않는다.
@WebServlet("/send")
public class SendServlet extends HttpServlet {// 1. HttpServlet 클래스를 상속받는다.

  // 2. service() 메소드 오버라이드
  @Override
  protected void service(HttpServletRequest req, HttpServletResponse resp) throws ServletException, IOException{
    System.out.println("요청이 옵니다."); //콘솔에 출력

    //  요청을 한 클라이언트에게 문자열을 응답할 객체 얻어오기
    PrintWriter pw = resp.getWriter();
    //웹브라우저에 출력하기
    pw.println("Oh my gosh!");
    pw.println("what the heck?");
    pw.flush();//방출
    pw.close();//닫아주기
  }
}
```

- 한글 콘솔에서 받을 수 있게 만들기

```
req.setCharacterEncoding("utf-8");
```

- 전송되는 문자열을 읽어오기

```
String a = req.getParameter("msg");
```

- 응답 인코딩 설정

```
resp.setCharacterEncoding("utp-8");
```

- 응답 컨텐츠 설정

```
resp.setContentType("text/html; charset=utf-8");
```

```
package test.servlet;

import java.io.IOException;
import java.io.PrintWriter;

import javax.servlet.ServletException;
import javax.servlet.annotation.WebServlet;
import javax.servlet.http.HttpServlet;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;

// 3. 요청 맵핑( 처리할 요청 경로를 작성한다) 반드시 /로 시작을 해야 하고 프로젝트명은 쓰지 않는다.
@WebServlet("/send")
public class SendServlet extends HttpServlet {// 1. HttpServlet 클래스를 상속받는다.

  // 2. service() 메소드 오버라이드
  @Override
  protected void service(HttpServletRequest req, HttpServletResponse resp) throws ServletException, IOException{
    System.out.println("요청이 옵니다."); //콘솔에 출력
    req.setCharacterEncoding("utf-8");

    //msg라는 파라미터명으로 전송되는 문자열 읽어오기
    String a = req.getParameter("msg");
    System.out.println("msg : "+a);

    //응답 인코딩 설정
    resp.setCharacterEncoding("utp-8");
    //응답 컨텐츠 설정
    resp.setContentType("text/html; charset=utf-8");

    //  요청을 한 클라이언트에게 문자열을 응답할 객체 얻어오기
    PrintWriter pw = resp.getWriter();
    //웹브라우져에 출력하기
    pw.println("<!doctype html>");
    pw.println("<html>");
    pw.println("<head>");
    pw.println("<meta charset='utf-8'>");
    pw.println("<title> 제목입니다.</title>");
    pw.println("</head>");
    pw.println("<body>");
    pw.println("<p>Oh my gosh!</p>");
    pw.println("<p>what the heck?</p>");
    pw.println("<img src='/Hello/images/redbird.png'>");
    pw.println("<img src=\"/Hello/images/yellowbird.png\">");
    pw.println("<br>");
    pw.println("<a href=\"/Hello/\">인덱스로 가기</a>");
    pw.println("<p>이것들은 뭐야?</P>");
    pw.println("</body>");
    pw.println("</html>");
    pw.flush();//방출
    pw.close();//닫아주기
  }
}
```