Spring MVC

13 서블릿과 JSP (1)

서블릿과 JSP

1. 서블릿과 컨트롤러의 비교

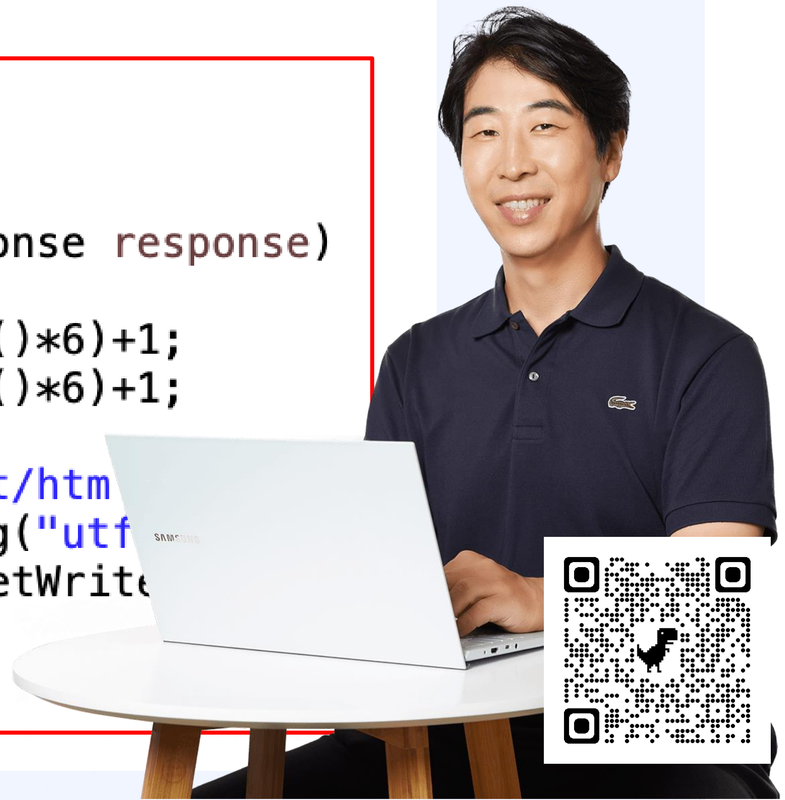
```java
@WebServlet("/rollDice2")
public class TwoDiceServlet extends HttpServlet {
    @Override
    public void service(HttpServletRequest request, HttpServletResponse response)
            throws IOException {
        int idx1 = (int)(Math.random()*6)+1;
        int idx2 = (int)(Math.random()*6)+1;

        response.setConte
        response.setChara
        PrintWriter out =
        out.println("<htm
        out.println("<hea
        out.println("</he
        out.println("<bod
        out.println("<img
        out.println("<img
        out.println("</bo
        out.println("</ht
        out.close();
    }
}
```

```java
@Controller
public class TwoDice {
    @RequestMapping("/rollDice")
    public void main(HttpServletResponse response)
            throws IOException {
        int idx1 = (int)(Math.random()*6)+1;
        int idx2 = (int)(Math.random()*6)+1;

        response.setContentType("text/htm
        response.setCharacterEncoding("utf
        PrintWriter out = response.getWrite

        // ...
```

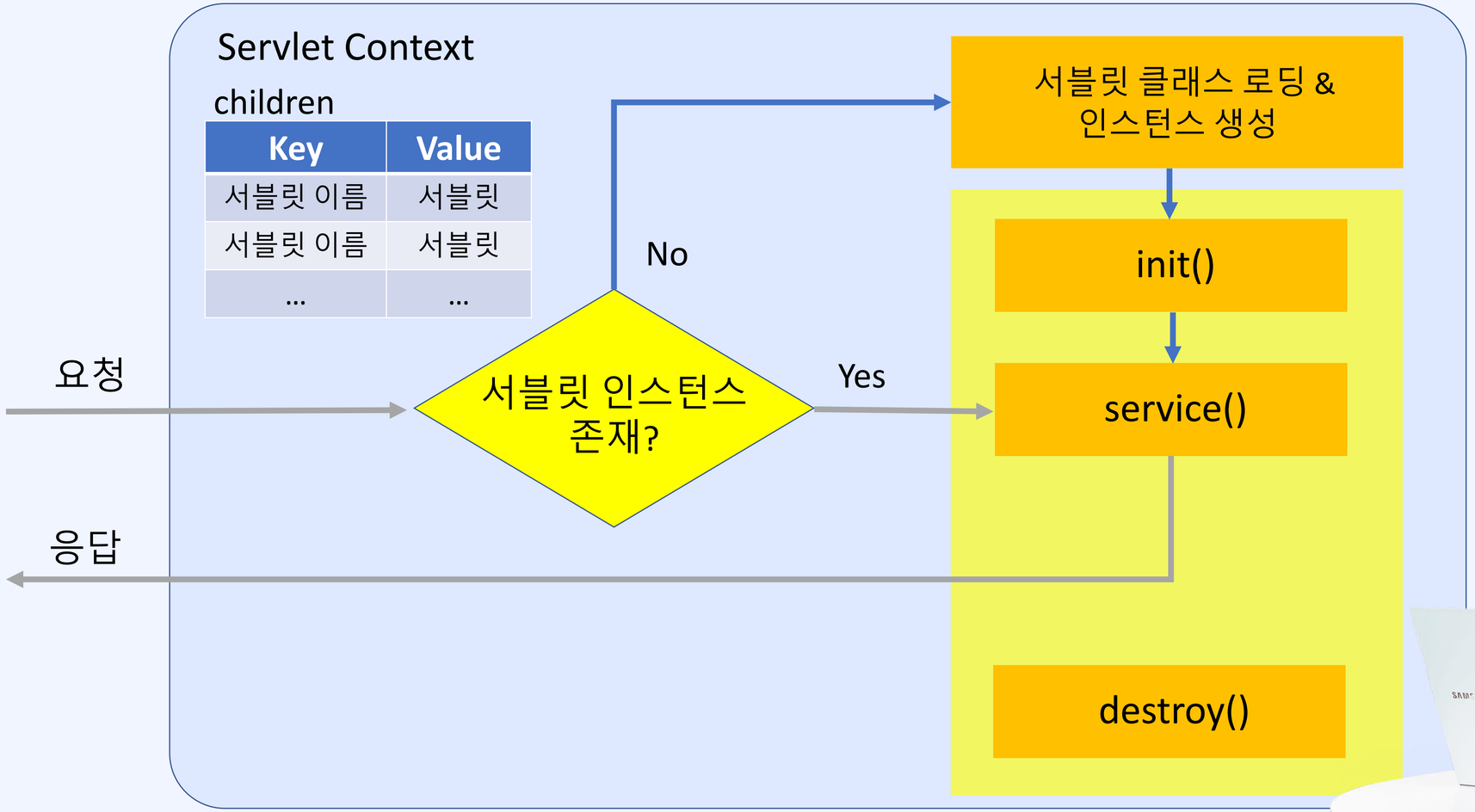


```java
@WebServlet("/hello")
public class HelloServlet extends HttpServlet {
    @Override
    public void init() throws ServletException {
        // 서블릿 초기화 - 서블릿이 생성 또는 리로딩 때, 단 한번만 수행됨.
        System.out.println("[HelloSerlvet] init()");
    }

    @Override // 호출될 때마다 반복적으로 수행됨.
    public void service(HttpServletRequest request, HttpServletResponse response) {
        // 1. 입력
        // 2. 처리
        // 3. 출력
        System.out.println("[HelloSerlvet] service()");
    }

    @Override
    public void destroy() {
        // 뒷정리 작업 - 서블릿이 제거(unload)될 때, 단 한번만 수행됨.
        System.out.println("[HelloSerlvet] destroy()");
    }
}
```

서블릿과 JSP
2. 서블릿의 생명주기



서블릿과 JSP

4. JSP와 서블릿의 비교

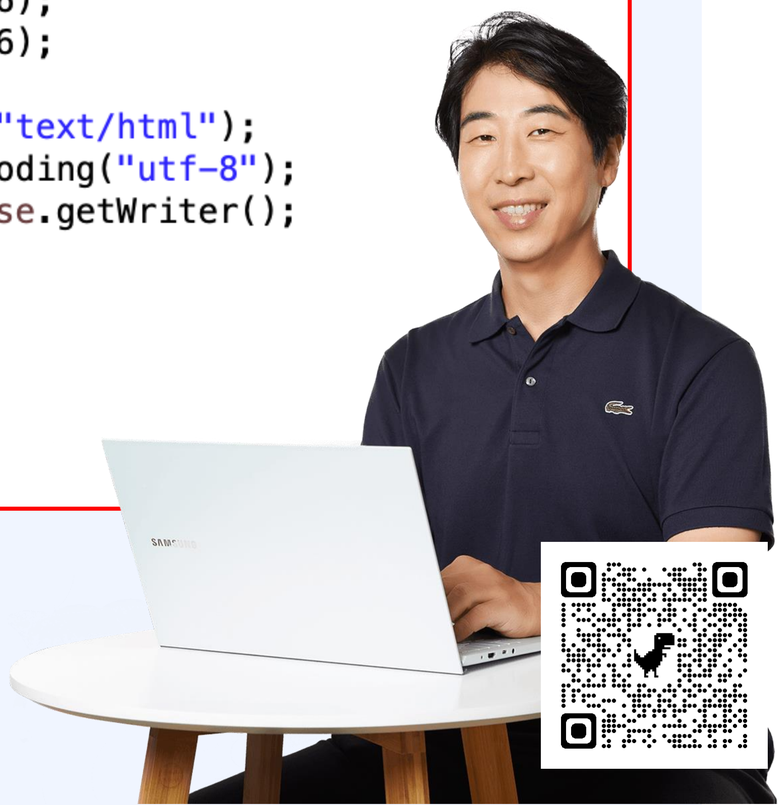
[src/main/webapp/twoDice.jsp] - http://loc

```jsp
<%@ page contentType="text/html;charset=
<%@ page import="java.util.Random" %>
<%-- <%! 클래스 영역 %> --%>
<%!
    int getRandomInt(int range){
        return new Random().nextInt(ran
    }
%>
<%-- <%  메서드 영역 - service()의 내부 %> -
<%
    int idx1 = getRandomInt(6);
    int idx2 = getRandomInt(6);
%>
<html>
<head>
    <title>twoDice.jsp</title>
</head>
<body>
    <img src='resources/img/dice<%=idx1%>.jpg'>
    <img src='resources/img/dice<%=idx2%>.jpg'>
</body>
</html>
```

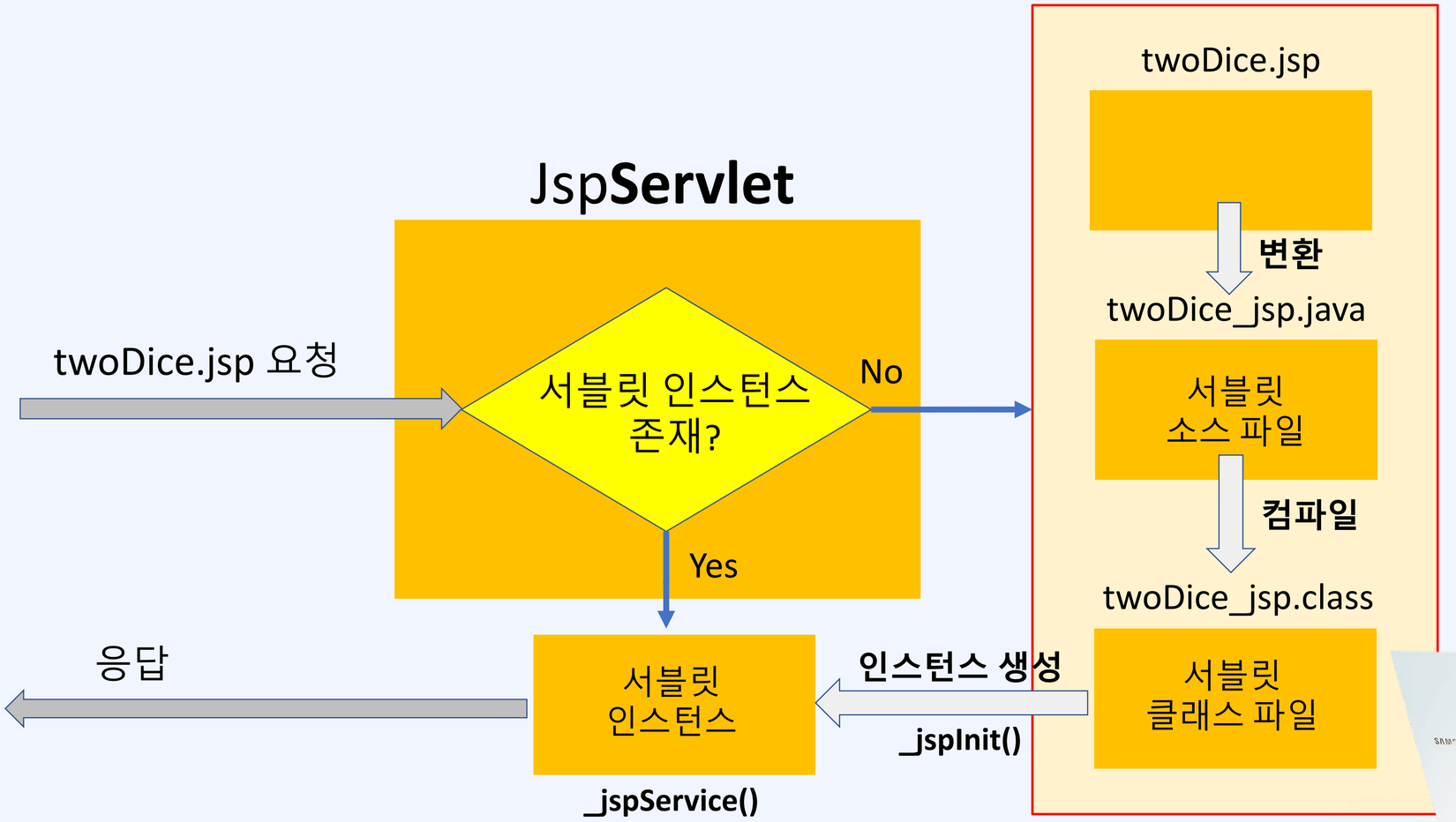
```java
@WebServlet("/rollDice2")
public class TwoDiceServlet extends HttpServlet {
    int getRandomInt(int range) {
        return new Random().nextInt(range)+1;
    }

    public void service(HttpServletRequest request,
            HttpServletResponse response) throws IOException {
        int idx1 = getRandomInt(6);
        int idx2 = getRandomInt(6);

        response.setContentType("text/html");
        response.setCharacterEncoding("utf-8");
        PrintWriter out = response.getWriter();
        out.println("<html>");
        out.println("<head>");
        out.println("</head>");
        // ...
    }
}
```



서블릿과 JSP
5. JSP의 호출 과정



[스프링의 정석 - 기초편] 남궁성과 끝까지 간다. fastcampus.co.kr

서블릿과 JSP

6. JSP와 서블릿으로 변환된 JSP의 비교

[src/main/webapp/twoDice.jsp]

```
<%@ page contentType="text/htm
<%@ page import="java.util.Ran
<%-- <%! 클래스 영역 %> --%>
<%!
    int getRandomInt(int range
        return new Random().ne
    }
%>
<%-- <%  메서드 영역 - service()의
<%
    int idx1 = getRandomInt(6)
    int idx2 = getRandomInt(6)
%>
<html>
<head>
    <title>twoDice.jsp</title>
</head>
<body>
    <img src='resources/img/di
    <img src='resources/img/dice<%=idx2%>.jpg'>
</body>
</html>
```

```
public final class twoDice_jsp extends org.apache.jasper.runtime.HttpJspBase
    implements org.apache.jasper.runtime.JspSourceDependent,
                org.apache.jasper.runtime.JspSourceImports {

        int getRandomInt(int range){
            return new Random().nextInt(range)+1;
        }

    public void _jspService(final javax.servlet.http.HttpServletRequest request,
                            final javax.servlet.http.HttpServletResponse response)
      throws java.io.IOException, javax.servlet.ServletException {
// ...
    int idx1 = getRandomInt(6);
    int idx2 = getRandomInt(6);

      out.write("\t<html>\n");
      out.write("\t<head>\n");
      out.write("\t\t<title>twoDice.jsp</title>\n");
      out.write("\t</head>\n");
      out.write("\t<body>\n");
      out.write("\t\t<img src='resources/img/dice
      out.print(idx1);
      out.write(".jpg'>\n");
// ...
```

서블릿과 JSP
7. JSP의 기본 객체

```jsp
<%@ page contentType="text/html
<%@ page import="java.util.Cale
<%
    String year  = request.getP
    String month = request.getP
    String day   = request.getP

    int yyyy = Integer.parseInt
    int mm   = Integer.parseInt
    int dd   = Integer.parseInt

    // 2. 처리
    Calendar cal = Calendar.get
    cal.set(yyyy, mm - 1, dd);

    int dayOfWeek = cal.get(Cale
    char yoil = " 일월화수목금토".charAt(dayOfWeek);
%>
<html>
<head>
    <title>yoilTeller.jsp</title>
</head>
<body>
    <h1><%=yyyy%>년 <%=mm%>월 <%=dd%>일은 <%=yoil%>요일입니다.</h1>
</body>
</html>
```

```java
public final class yoilTeller_jsp extends org.apache.jasper.runtime.HttpJspBase
    implements org.apache.jasper.runtime.JspSourceDependent,
                 org.apache.jasper.runtime.JspSourceImports {
// ...

    public void _jspService(final javax.servlet.http.HttpServletRequest request,
                            final javax.servlet.http.HttpServletResponse response)
      throws java.io.IOException, javax.servlet.ServletException {
// ...
    final javax.servlet.jsp.PageContext pageContext;
    javax.servlet.http.HttpSession session = null;
    final javax.servlet.ServletContext application;
    final javax.servlet.ServletConfig config;
    javax.servlet.jsp.JspWriter out = null;
    final java.lang.Object page = this;
//...
```

서블릿과 JSP
7. JSP의 기본 객체

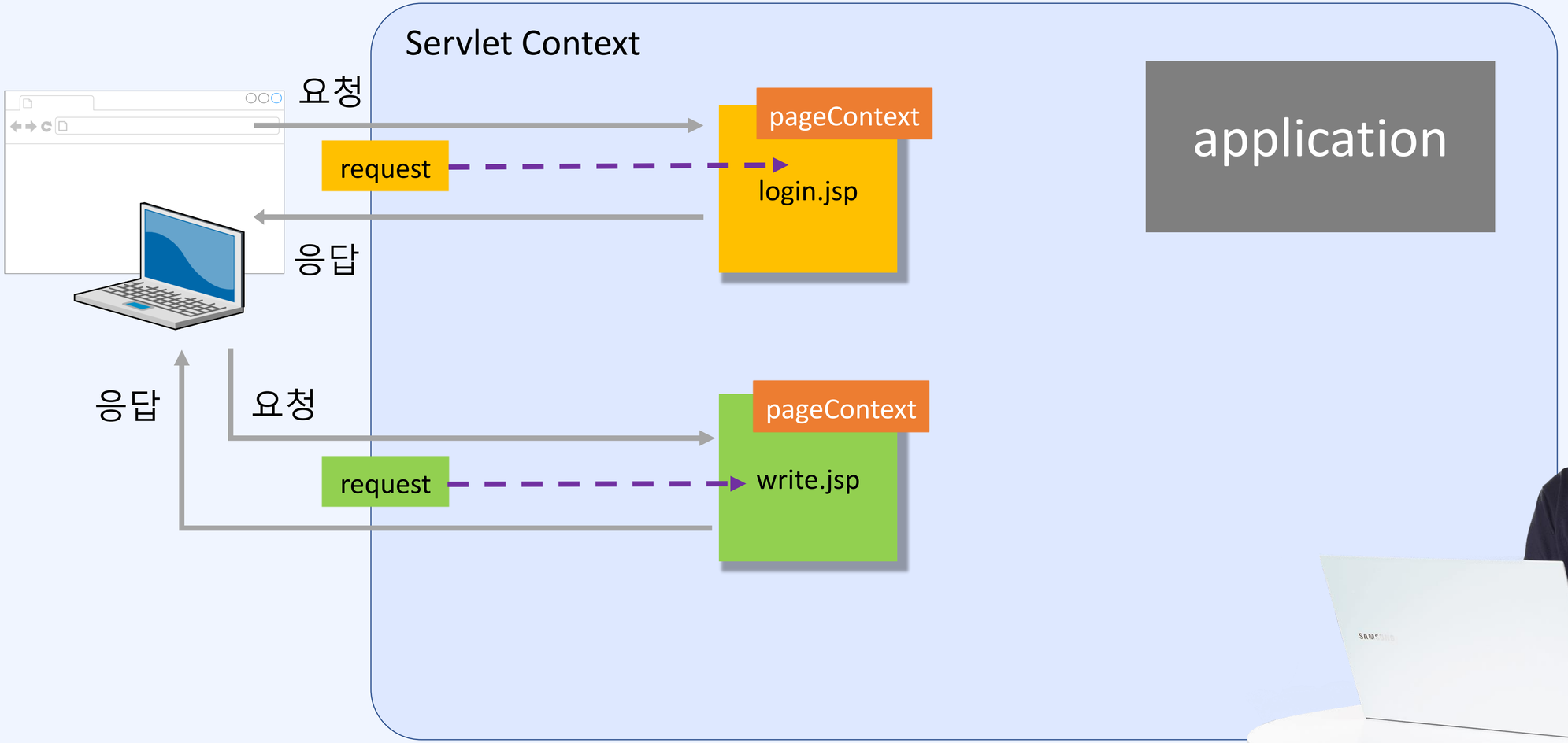
기본 객체	타입	설명
request	**javax.servlet.http.HttpServletRequest**	**요청 정보가 담겨있는 객체**
response	**javax.servlet.http.HttpServletResponse**	**요청에 응답을 작성할 때 사용**
session	**javax.servlet.http.HttpSession**	**HTTP session을 구현한 객체. 세션 정보 저장에 사용**
application	javax.servlet.ServletContext	Web Application 전체에서 공유하는 객체
config	javax.servlet.ServletConfig	JSP 페이지에 대한 설정 정보가 담긴 객체
page	java.lang.Object	JSP 페이지 객체 자신
pageContext	javax.servlet.jsp.PageContext	JSP페이지의 context정보를 제공
out	**javax.servlet.jsp.JspWriter**	**응답에 포함될 내용을 출력할 때 사용**
exception	java.lang.Throwable	예외가 발생했을 때 생성되는 예외 객체

Spring MVC

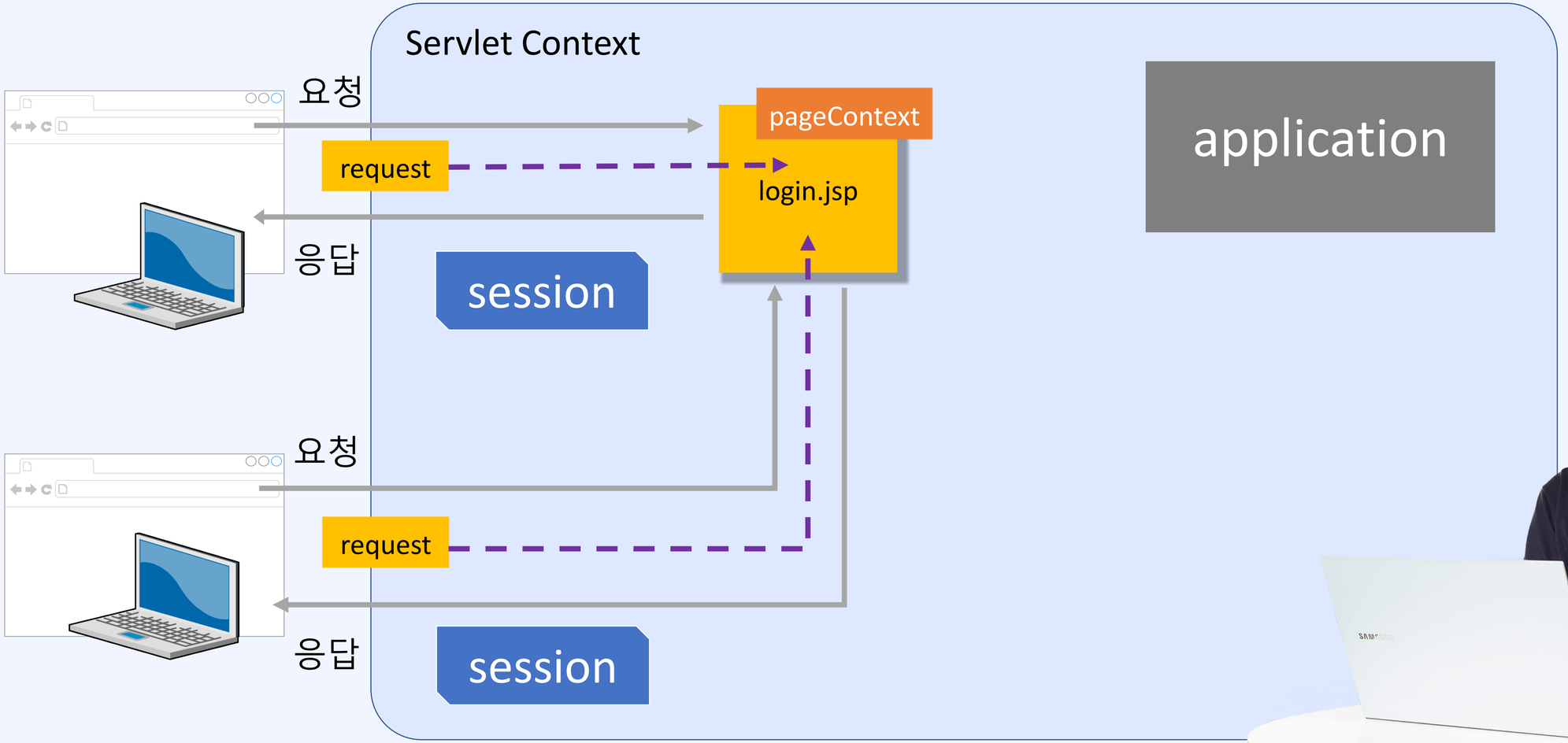
14 서블릿과 JSP (2)

서블릿과 JSP
8. 유효 범위(scope)와 속성(attribute)

Key	Value
속성1	속성값
속성2	속성값
...	...

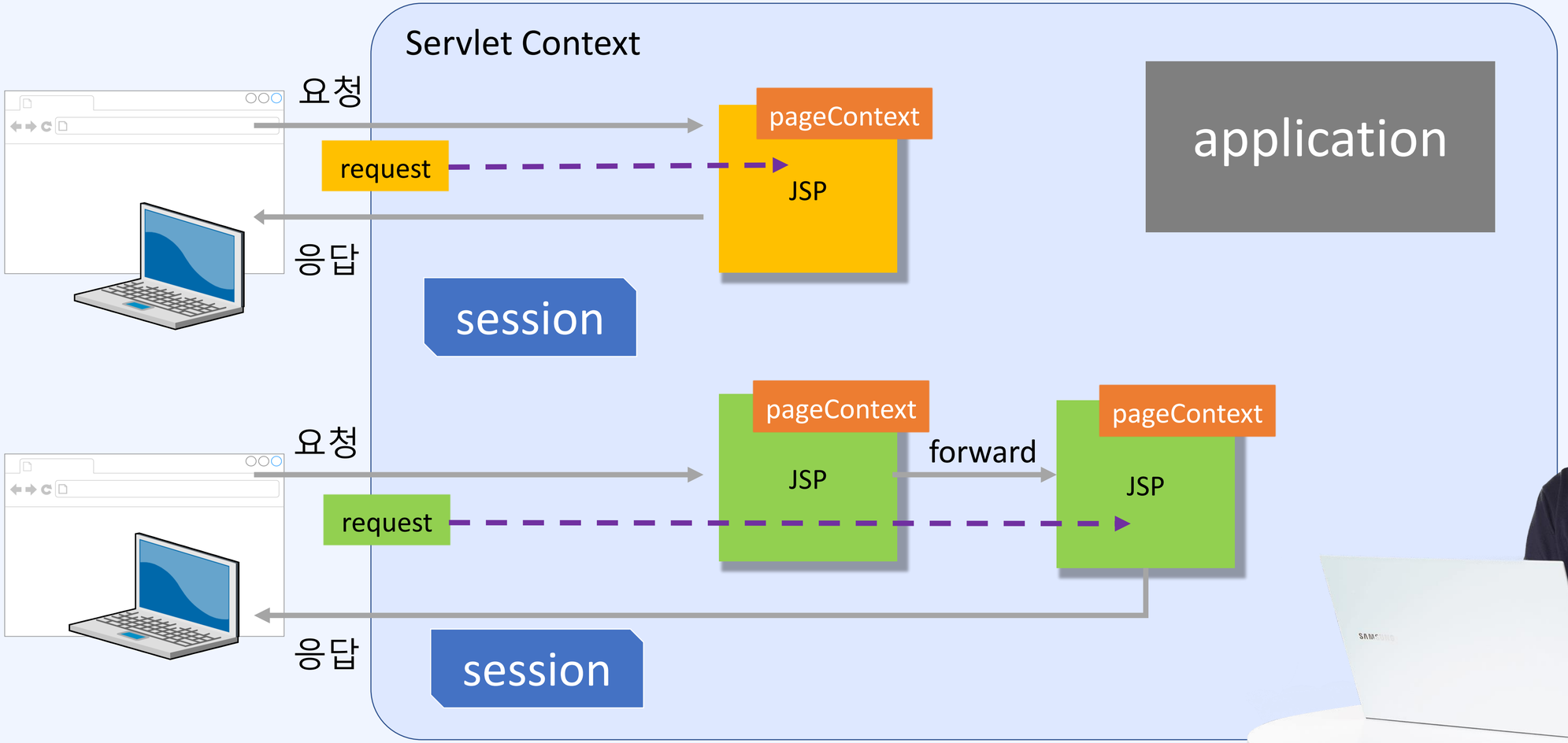


서블릿과 JSP
8. 유효 범위(scope)와 속성(attribute)



서블릿과 JSP
8. 유효 범위(scope)와 속성(attribute)

Key	Value
속성1	속성값
속성2	속성값
...	...



서블릿과 JSP
8. 유효 범위(scope)와 속성(attribute)

기본 객체	유효 범위	설명
pageContext	1개 JSP페이지	JSP페이지의 시작부터 끝까지. 해당 JSP 내부에서만 접근가능. 페이지당 1개
request	1+개 JSP페이지	요청의 시작부터 응답까지. 다른 JSP로 전달 가능. 요청마다 1개
session	n개 JSP페이지	session의 시작부터 종료까지(로그인 ~ 로그아웃). 클라이언트마다 1개
application	context 전체	Web Application의 시작부터 종료까지. context내부 어디서나 접근 가능 모든 클라이언트가 공유. context마다 1개

속성 관련 메서드	설명
void setAttribute(String name, Object value)	지정된 값(value)을 지정된 속성 이름(name)으로 저장
Object getAttribute(String name)	지정된 이름(name)으로 저장된 속성의 값을 반환
void removeAttribute(String name)	지정된 이름(name)의 속성을 삭제
Enumeration getAttributeNames()	기본 객체에 저장된 모든 속성의 이름을 반환

Spring MVC

15 서블릿과 JSP (3)

```
//@WebServlet(urlPatterns={"/hello", "/hello/*"}, loadOnStartup=1)
  @WebServlet("/hello")
  public class HelloServlet extends HttpServlet {
```


@WebServlet으로 서블릿을 URL에 맵핑할 때 사용

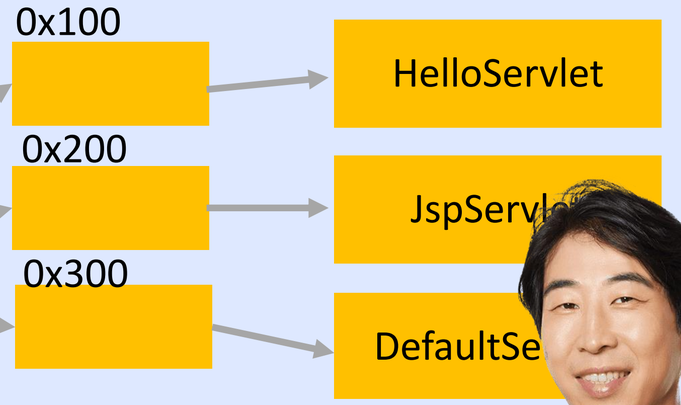
종류	URL pattern	매칭 URL
1. exact mapping	/login/hello.do	http://localhost/ch2/login/hello.do
2. path mapping	/login/*	http://localhost/ch2/login/ http://localhost/ch2/login/hello http://localhost/ch2/login/hello.do http://localhost/ch2/login/test/
3. extension mapping	*.do	http://localhost/ch2/hi.do http://localhost/ch2/login/hello.do
4. default mapping	/	http://localhost/ch2/ http://localhost/ch2/hello.do http://localhost/ch2/login/ http://localhost/ch2/login/hello http://localhost/ch2/login/hello.do

서블릿과 JSP
9. URL 패턴

Servlet Context

children(서블릿)

Key(String, 서블릿 이름)	Value(StandardWrapper)
com.fastcampus.ch2.HelloServlet	0x100
jsp	0x200
default	0x300
...	...



servletMappings

Key(String)	Value(String, 서블릿 이름)
/hello	com.fastcampus.ch2.HelloServlet
*.jsp	jsp
/	default
...	...

요청

서블릿과 JSP

10. EL (Expression Language)

```
<body>
person.getCar().getColor()=<%=person.getCar().getColor()%> <br>
person.getCar().getColor()=${person.getCar().getColor()} <br>
person.car.color=${person.car.color} <br>
name=<%=request.getAttribute("name")%> <br>
name=${requestScope.name} <br>
name=${name} <br>
id=<%=request.getParameter("id")%> <br>
id=${pageContext.request.getParameter("id")} <br>
id=${param.id} <br>
"1"+1 = ${"1"+1} <br>
"1"+="1" = ${"1"+="1"} <br>
"2">1 = ${"2">1} <br>
null = ${null}<br>
null+1 = ${null+1} <br>
null+null = ${null+null} <br>
"" + null = ${""+null} <br>
""-1 = ${""-1} <br>
empty null=${empty null} <br>
empty list=${empty list} <br>
null==0 = ${null==0} <br>
null eq 0 = ${null eq 0} <br>
name == "남궁성"=${name=="남궁성"} <br>
name != "남궁성"=${name!="남궁성"} <br>
name eq "남궁성"=${name eq "남궁성"} <br>
name ne "남궁성"=${name ne "남궁성"} <br>
name.equals("남궁성")=${name.equals("남궁성")} <br>
</body>
```

```
<%
    Person person = new Person();
    request.setAttribute("person", person);
    request.setAttribute("name", "남궁성");
    request.setAttribute("list", new java.util.ArrayList());
%>
```

Spring MVC

16 서블릿과 JSP (4)

서블릿과 JSP

11. JSTL (JSP Standard Tag Library)

```
<%@ taglib prefix="c"   uri="http://java.sun.com/jsp/jstl/core"%>
<%@ taglib prefix="fmt" uri="http://java.sun.com/jsp/jstl/fmt" %>
```

```
<c:set var="to"   value="10"/>
<c:set var="arr"  value="10,20,30,40,50,60,70"/>
<c:forEach var="i" begin="1" end="${to}">
    ${i}
</c:forEach>
<br>
<c:if test="${not empty arr}">
    <c:forEach var="elem" items="${arr}" varStatus="status">
        ${status.count}. arr[${status.index}]=${elem}<BR>
    </c:forEach>
</c:if>
<c:if test="${param.msg != null}">
    msg=${param.msg}
    msg=<c:out value="${param.msg}"/>
</c:if>
<br>
<c:if test="${param.msg == null}">메시지가 없습니다.<br></c:if>
<c:set var="age" value="${param.age}"/>
<c:choose>
    <c:when test="${age >= 19}">성인입니다.</c:when>
    <c:when test="${0 <= age && age < 19}">성인이 아닙니다.</c:when>
    <c:otherwise>값이 유효하지 않습니다.</c:otherwise>
</c:choose>
<br>
<c:set var="now" value="<%=new java.util.Date() %>"/>
Server time is <fmt:formatDate value="${now}" type="both" pattern="yyyy/MM/dd HH:mm:ss"
```

```
<%
    if(msg !=null) {
%>
        msg=${param.msg}
<%
    } else {
%>
        메시지가 없습니다.
<%
    }
%>
```

공통적인 요청 전처리와 응답 후처리에 사용. 로깅, 인코딩 등

1. 전처리
2. 처리
3. 후처리

1. 전처리
2. 처리
3. 후처리

1. 전처리
2. 처리
3. 후처리

Filter

1. 전처리
2. 서블릿 호출
3. 후처리

서블릿

2. 처리

2. 처리

2. 처리

공통적인 요청 전처리와 응답 후처리에 사용. 로깅, 인코딩 등

Filter1

1. 전처리
2. 필터 호출
3. 후처리

Filter2

1. 전처리
2. 서블릿 호출
3. 후처리

서블릿

2. 처리

```java
//필터를 적용할 요청의 패턴 지정 - 모든 요청에
@WebFilter(urlPatterns="/*")
public class PerformanceFilter implem
    @Override
    public void init(FilterConfig fil
        // 초기화 작업
    }

    @Override
    public void doFilter(ServletReque
            throws IOException, Servl
        // 1. 전처리 작업
        long startTime = System.curre

        // 2. 서블릿 또는 다음 필터를 호출
        chain.doFilter(request, respo

        // 3. 후처리 작업
        System.out.print("["+((HttpSe
        System.out.println(" 소요시간="
    }

    @Override
    public void destroy() {
        // 정리 작업
    }
}
```

```java
@WebServlet("/rollDice2")
public class TwoDiceServlet extends HttpServlet {
    public void service(HttpServletRequest request,
            HttpServletResponse response) throws IOException {
        // 1. 전처리 작업
        long startTime = System.currentTimeMillis();

        int idx1 = (int)(Math.random()*6)+1;
        int idx2 = (int)(Math.random()*6)+1;

        response.setContentType("text/html");
        response.setCharacterEncoding("utf-8");
        PrintWriter out = response.getWriter();
        out.println("<html>");
        out.println("<head>");
        out.println("</head>");
        // ...

        // 3. 후처리 작업
        System.out.print("["+((HttpServletRequest)request).getRequestURI()+"]");
        System.out.println(" 소요시간="+(System.currentTimeMillis()-startTime)+"ms");
    }
}
```